(19) **KOREAN INTELLECTUAL PROPERTY OFFICE**

KOREAN PATENT ABSTRACTS

(11)Publication number: **1020050036702 A**

(43)Date of publication of application: **20.04.2005**

(21)Application number: **1020040070748**

(22)Date of filing: **06.09.2004**

(71)Applicant: **CANON KABUSHIKI KAISHA**

(72)Inventor: **CREW LAURENCE**
KING ANDREW JOHN
WHITFIELD
LO ANDREW JAMES
PATRICK LACHLAN
JAMES
TONISSON ALAN
VALEV

(51)Int. Cl **G06F 3/12**
G06F 17/27

(54) USER INTERFACE FOR GENERATING/EDITING/PRINTING VARIABLE DATA DOCUMENTS FORMED WITH TEXTS AND GRAPHICS

(57) Abstract:

PURPOSE: A user interface for generating/editing/printing variable data documents formed with texts and graphics is provided to simplify generation of a document template by enabling a user to edit constraints on a spot with, as the user clicks various positions on containers without switching to a separate screen or an area. CONSTITUTION: The document template is searched in a GUI (Graphic User Interface) on a display and has a start point(624). Container drawn by a user is detected through motion of a mouse pointer(626). A range of the container is changed by controlling one control point(634). Additional constraints are applied to a layout by the user until the layout is completed(638). After the layout is completed, records of the contents are arranged to the layout in order to generate the document. If all records are consumed and the document is generated, verification and/or print are performed.

Legal Status

Date of request for an examination (20040906)

Notification date of refusal decision (00000000)

Final disposal of an application (registration)

Date of final disposal of an application (20070409)

Patent registration number ( )

Date of registration (00000000)

Number of opposition against the grant of a patent ( )

Date of opposition against the grant of a patent (00000000)

Number of trial against decision to refuse ( )

Date of requesting trial against decision to refuse ( )

(19)대한민국특허청(KR)
(12) 공개특허공보(A)

(51) 。 Int. Cl.[7]
G06F 3/12
G06F 17/27

(11) 공개번호 10-2005-0036702
(43) 공개일자 2005년04월20일

(21) 출원번호 10-2004-0070748
(22) 출원일자 2004년09월06일

(30) 우선권주장 2003905659 2003년10월15일 오스트레일리아(AU)

(71) 출원인 캐논 가부시끼가이샤
일본 도쿄도 오오따꾸 시모마루꼬 3쪼메 30방 2고

(72) 발명자 로앤드류제임스
호주 뉴 사우쓰 웰즈 2040 레이챠드트 다니엘 스트리트 16
크루로렌스
호주 뉴 사우쓰 웰즈 2010 달링허스트 옥스포드 스트리트 82/6-14
토니슨알랜밸레브
호주 뉴 사우쓰 웰즈 2153 볼크햄 힐즈 더 코텔 웨이 30/3
킹앤드류존위트필드
호주 뉴 사우스 웰즈 2078 워룽가 폭스 밸리 로드 28
패트릭래칠랜제임스
호주 뉴 사우쓰 웰즈 2121 에핑 브릿지 스트리트 9/19

(74) 대리인 장수길
이중희
구영창

심사청구 : 있음

(54) 가변 데이터 문서들의 생성 및 편집을 위한 사용자인터페이스

요약

가변 문서(variable document) 프린팅을 위한 방법 및 장치들이 개시되는데, 본 방법 및 장치에서 그래픽 사용자 인터페이스는 가변 문서 생성을 위한 템플릿 내의 콘텐츠 콘테이너들(content containers)과 연관된 레이아웃 규칙들을 사용자가 조작할 수 있도록 구성된다. 한 가지 방법은 템플릿에 기초하여 가변 데이터 문서에 대한 레이아웃을 생성하는 단계를 포함한다. 이러한 방법은 우선 레이아웃을 형성하기 위해 템플릿 내에 적어도 하나의 콘테이너를 세팅하고, 이어서 상기 콘테이너의 적어도 하나의 특징 중 각각의 선택된 특징과 연관된 적어도 하나의 제한을 설정하며, 상기 설정 단계는 각각의 특징에 대해, 대응하는 특징의 사용자 유발 선택(user instigated selection)을 검출하는 단계를 포함한다. 이어서 콘텐츠를 콘테이너들에 배치함으로써 문서를 생성하도록 레이아웃이 변경되며, 레이아웃에서의 각각의 제한이 만족되는 것을 조건으로 배치된 콘텐츠의 속성에 기초하여, 레이아웃 내의 최소 하나의 콘테이너의 적어도 하나의 차원 및/또는 적어도 하나의 콘테이너의 위치가 변경된다.

대표도

도 20

색인어

그래픽 사용자 인터페이스, 가변 문서, 템플릿, 콘테이너, 레이아웃

명세서

도면의 간단한 설명

본 발명의 적어도 하나의 실시예가 이하에서 도면들을 참조하여 기술될 것이다.

도 1a는 가변 데이터 프린팅(variable data printing)을 위한 컴퓨터 시스템 구성을 도시하는 도면.

도 1b는 도 1a의 컴퓨터 모듈의 개략적인 블록도.

도 2는 가변 데이터 프린팅을 위한 대안적인 컴퓨터 시스템 구성을 도시하는 도면.

도 3은 메뉴 바들, 툴바들, 작업 영역(work area) 및 플로팅 팔레트(floating pallette)를 포함하는 애플리케이션 메인 윈도우의 예를 도시하는 도면.

도 4는 콘테이너들, 앵커들 및 슬라이더들 간의 스트럿들(struts)의 제 1 타입인 예시적인 콘테이너 생성을 포함하는 본 명세서의 태양들을 보여주기 위해 스크린, 툴들 및 아이콘들을 도시하는 도면.

도 5a 내지 도 5d는 제1 예시적인 콘테이너 규칙들을 도시하는 도면.

도 6a 내지 도 6c는 제3 콘테이너 규칙들을 도시하는 도면.

도 6d 내지 도 6e는 콘테이너 범위들(container extents)의 생성을 묘사하는 흐름도.

도 7a 내지 도 7b는 세 개의 열을 갖는 텍스트 콘테이너들과, 행 및 거터 폭(gutter width)들 모두가 직접적인 조작에 의해 포인팅 디바이스를 사용하여 리사이징될 수 있는 방법을 도시하는 도면.

도 8은 문자 크기 동기화의 동작을 보여주는 스크린, 툴들 및 아이콘들을 도시하는 도면.

도 9는 내부 가장자리들의 자동 적용(automatic application) 동작을 보여주는 스크린, 툴들 및 아이콘들을 도시하는 도면.

도 10은 거리 제한을 기술하기 위해 두 개의 콘테이너들의 모서리들 간에 스트럿이 추가될 수 있는 방법을 도시하는 도면.

도 11은 제2 예시적인 콘테이너들 사이에 스트럿들을 포함하는 사용자 인터페이스의 몇가지 특징들의 동작을 보여주는 스크린, 툴들 및 아이콘들을 도시하는 도면.

도 12는 비고정된 가이드들의 동작을 보여주는 스크린을 도시하는 도면.

도 13 및 도 14는 데이터 소스 선택 방법을 도시하는 도면.

도 15는 데이터 소스 필터링을 위한 사용자 인터페이스를 도시하는 도면.

도 16 및 도 17은 데이터 소스를 통해 네비게이팅하기 위한 방법을 보여주는 사용자 인터페이스를 도시하는 도면.

도 18은 데이터 소스의 변수들을 보여주는 사용자 인터페이스의 일례와 데이터 소스를 통해 네비게이팅하기 위한 방법을 도시하는 도면.

도 19는 가변 문서 템플릿과 데이터 소스의 변수를 연관시키기 위한 사용자 인터페이스를 도시하는 도면.

도 20은 가변 문서 템플릿과 병합된 데이터 소스 콘텐츠의 실제와 유사한 증명을 위한 사용자 인터페이스를 도시하는 도면.

도 21a 및 도 21b는 선택적인 증명에 사용하기 위해, 평균적인 문서 및 각각의 문서에 대한 콘테이너들의 폭들 및 높이들을 사용하는 가장 특별한 문서를 계산하기 위한 방법을 도시하는 도면.

도 22는 레이아웃 방법을 위한 입력으로 사용되는 레이아웃 아이템들 및 제한들을 보여주는 예시적인 레이아웃.

도 23은 도 22에서의 예에 대응하는 수직 제한들(vertical constraints)만을 도시하는 도면.

도 24는 도 23의 수직 제한들을 나타내는 방향 그래프(directed graph).

도 25는 푸쉬 동작을 수행하는 단계에 포함된 단계들을 도시하는 도면.

도 26은 푸쉬 동작의 대안적인 구현에 포함된 단계들을 도시하는 도면.

도 27a 내지 도 27f는 도 27a 및 도 27b가 푸쉬 동작의 예의 시작 위치를 도시하고, 도 27c 및 도 27d가 푸쉬 동작의 예의 중간 단계를 도시하고, 도 27e 및 도 27f는 푸쉬 동작의 예의 결과를 도시하는, 사용 시의 푸쉬 동작의 예를 도시하는 도면.

도 28은 그래프 기반의 레이아웃 계산(graph basd layout calculation)에서의 고수준 단계들을 도시하는 도면.

도 29는 그래프 기반의 레이아웃 계산의 구현에서의 고수준 단계들을 도시하는 도면.

도 30a는 그래프 기반 레이아웃 계산이 수직 마크들을 이동시킴으로써 에너지 함수의 값을 어떻게 감소시키는지를 도시하는 도면.

도 30b는 고정된 중심 규칙들(fixed center rules)을 처리하는 단계를 포함하는 그래프 기반 레이아웃 방법의 또 다른 구현을 도시하는 도면.

도 31은 그래프 기반 레이아웃 계산에서 한 그룹의 마크들이 이동하는 거리가 어떻게 계산되는 지를 도시하는 도면.

도 32는 한 세트의 마크들이 우측으로 푸쉬된 경우에 푸쉬된 마크들의 앞쪽 세트들 및 마주보는 세트들을 계산하는 단계에 포함된 단계들을 도시하는 도면.

도 33a 내지 도 33c는 콘테이너의 최소 및 최대 높이가 기초 모델(basic model)의 다른 표현들을 사용하여 어떻게 지정될 수 있는지를 도시하는 도면.

도 34는 레이아웃 내의 콘테이너의 에지 위치 및 다른 에지들과의 상호작용을 변경하는 단계의 흐름도.

도 35는 템플릿 문서들을 편집하기 위한 푸쉬 동작의 또 다른 예의 흐름도.

도 36은 가변 데이터 문서들을 생성하고 프린트하는 방법의 흐름도.

도 37a 내지 도 37d는 도 4에 대응하는 예시적인 레이아웃 및 레이아웃의 다양한 제한들이 저장될 수 있는 방법을 도시하는 도면.

도 38은 제2 콘테이너 규칙들에서 콘테이너 제한들을 나타내기 위해 높이 및 폭 바들의 사용을 도시하는 도면.

도 39는 1차원으로 동작하는 레이아웃 엔진의 예시적인 사용을 도시하는 도면.

도 40a 및 도 40b는 2차원으로 동작하는 레이아웃 엔진의 예시적인 사용을 도시하는 도면.

도 41a 및 도 41k는 콘테이너 내에서 텍스트를 레이아웃하는 방법과 이러한 방법이 콘테이너 모양에 의해 어떻게 영향을 받을 수 있는지를 묘사하는 도면.

도 42a 및 도 42c는 테이블들의 구성을 위한 접근법들을 설명하는 도면.

* 도면의 주요 부분에 대한 부호의 설명 *

101: 호스트 컴퓨터

121: 레이아웃 편집 애플리케이션

103: 사용자 인터페이스

105: 레이아웃 엔진

107: 네트워크 접속

117: 데이터베이스 서버

119: 데이터베이스

115: 파일 서버

109: 프린트 서버

113: 프린터

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

관련 특허 출원들에 대한 교차 참조

본 출원은 2003년 10월 15일자로 출원된 오스트레일리아 특허 출원번호 제 2003905659호에 기초하여 35 U.S.C §119 하에서 우선권을 주장하며, 본 명세서에서 완전히 제시되는 것과 같이 그 전체가 본 명세서에 참조에 의해 포함된다.

본 발명은 텍스트 및 그래픽으로 구성된 문서들의 생성, 편집, 및 프린트를 위해 소프트웨어로 구현된 방법 및 프로세스들에 관한 것이며, 특히 가변 데이터 문서들의 생성, 편집, 및 프린트에 관한 것이다.

가변 데이터 프린팅

통상적으로, 다수의 문서들을 프린트하는 경우에 사용되는 프린팅 프로세스는 문서들의 물리적 표현물을 요구하는 아날로그 프로세스였다. 이들은 프린트 전에 생성되어야하는 인쇄판들(printing plates) 및/또는 브로마이드(bromide)들을 포함한다. 이는, 맞춤 문서들(customized documents)을 제작하는 데 많은 비용이 소모되고, 완전히 동일한 문서의 다수의 복사본을 프린트하는 것을 포함하는 대량의 문서를 인쇄한느데 많은 비용이 소모됨을 의미한다.

맞춤 문서들은 문서의 물리적 복사본을 프린트 전에 준비할 필요가 없는 디지털 프린팅 프로세스를 이용하여 생성될 수 있다. 최근까지, 디지털 프로세스들은 비용이 많이 들거나, 그 결과물들이 통상의 아날로그 프린팅 프로세스들에 의해 생성된 것보다 더 열악한 품질이었다.

그러나, 최근 5년간, 디지털 프린팅의 진보로 인해 고품질 맞춤 문서들을 제작하기 위한 비용이 상당히 감소되었다. 이러한 진보들로 인해 가변 데이터 프린트가 점차로 인기있게 되었다. 문서 작성자들이 단지 하나의 문서가 아니라 관련된 일련의 문서들을 생성할 수 있도록 하는 가변 데이터 프린팅 소프트웨어가 이용가능하게 되었고, 이 경우 각각의 문서는 의도된 독자들을 위해 맞춤제작된다.

고품질 맞춤 프린팅을 실행가능하게 하기 위해 몇가지 문제들이 해결되어야할 필요가 있다. 한가지 문제는 프린트된 출력물의 품질이다. 최근에 들어서야 디지털 프로세스들은 전통적인 프린팅 프로세스들의 해상도에 접근할 수 있었다. 또 다른 문제는 디지털 형태로 고품질 문서를 나타내기 위하여, 특히 문서가 고해상도 이미지들을 포함하는 경우에, 통상적으로 다량의 데이터가 요구된다는 점이다. 이로 인해, 다수의 문서들을 컴퓨터 네트워크를 통해 디지털 프린터로 전기적으로 전송하는 것이 비실용적이 된다. 세번째 문제는, 통상적으로 문서들이 간단한 문서들보다 생성 및 유지에 더 많은 노력과 기술을 요구한다는 점이다. 문서들은 가변 데이터(variable data)를 문서 템플릿에 삽입함으로써 생성된다. 문서 템플릿은 가변 데이터를 삽입하기 위한 슬롯들을 갖는 문서이다. 문서 템플릿은 많은 문서들에 공통되는 일부 데이터, 및 템플릿의 가변 데이터 슬롯에 데이터를 삽입함으로써 생성되는 각각의 맞춤 문서를 레이아웃하는 방법을 정의하는 정보를 포함할 수 있다. 페이지들의 레이아웃은 데이터의 크기 변화 또는 데이터 아이템 분실에 대응할 수 있도록 정의될 필요가 있기 때문에, 가변 데이터 템플릿을 만드는 것은 간단한 문서를 작성하는 것보다 더 어렵다. 또한, 템플릿과 데이터 간의 관계가 정의 및 유지될 필요가 있다. 이러한 경우에, 종종 가변 데이터 프린팅 기술을 사용하여 문서들을 생성하기 위해 데이터베이스 기술들이 요구된다.

디지털 프린팅 하드웨어의 품질은, 결과물들이 하이-엔드(high-end) 아날로그 프린트 장치에 의해 생성된 것과 비교할 수 있을 정도로 개선된 반면, 비용은 무척 낮아졌다. PPML과 같은 새로운 표준들로 인해, 대량의 반복 데이터를 전송하지 않고도 관련된 문서의 그룹들을 네트워크를 통해 프린터로 송신할 수 있게 됐다. 이로 인해 가변 데이터 프린팅을 위해 요구되는 네트워크 대역폭이 감소되었다.

가변 데이터 문서들의 작성 및 관리의 복잡성이라는 문제에 대한 어떠한 손쉬운 해결책도 존재하지 않는다. 수 년간, 프린팅은 "메일-통합(mail-merge)"의 형태로 워드 프로세싱 소프트웨어에서 조잡한 형태로 이용가능했지만, 메일 통합은 개별 출력 문서들의 레이아웃에 대한 아주 조잡한 제어만을 지원한다. 가장 최근의 가변 데이터 프린팅 소프트웨어는, 각각의 문서에서 상이할 수 있는 데이터 아이템들의 크기 및 모양을 수용하는 정교하고 유연성있는 레이아웃들의 생성을 허용하지만, 이러한 애플리케이션들은 여전히 사용하기에 복잡하고 어렵다.

레이아웃 모델들

통상적으로, 텍스트 및 그래픽 모두를 포함하는 문서는 상이한 타입의 아이템들의 집합으로서 구성된다. 아이템들은 텍스트, 그래픽 또는 다른 종류의 추상적 개체 - 아이템들의 집합인 그룹과 같은 것 - 일 수 있다. 레이아웃 아이템들은 통상적으로 직사각형 모양을 가지거나 직사각형 범위(extent)를 갖는다. 아이템들을 레이아웃하는 방법을 정의하기 위하여 각각의 애플리케이션에 의해 허용된 규칙들이 레이아웃 모델을 나타낸다. 가변 데이터 프린팅 애플리케이션에서, 상기 규칙들은 아이템들의 크기가 변화함에 따라 레이아웃이 계산되는 방법을 정의해야한다.

계층적 레이아웃 모델들(Hierarchical Layout Models)

현재 가변 데이터 프린팅 솔루션들은 통상적으로 레이아웃을 정의하기 위해 계층적 모델들을 사용한다. 이러한 모델들은 HTML 및 XML 문서용으로 사용되는 것들과 유사하다. 이러한 모델들의 예로서 CSS 박스 모델 및 XSL:FO가 있다. 이러한 레이아웃 모델들에서, 레이아웃 영역 내부에 수직 또는 수평으로 직사각형 아이템들을 스택킹함으로써(stacking) 직사각형 아이템들이 직사각형 레이아웃 영역에서 레이아웃된다.

수평 및 수직 스택킹에 부가하여, 일부 애플리케이션들에 의해 지원되는 또다른 보다 일반적인 대안은, 페이지 상의 단어들의 행처럼, 아이템들이 수직 열들 또는 수평 행들로 스택킹되도록 하는 것이다. 이러한 형태의 레이아웃에서, 행 또는 열

에 들어가야 할 너무 많은 아이템들이 존재하면, 새로운 행 또는 열이 시작되고 초과 아이템들은 열의 다음 행으로 "랩(wrap)"한다. 이는 워드 프로세서에서의 단어 랩핑(word wrapping)과 유사하다. 아이템들을 레이아웃하는 이러한 방법은 웹 애플리케이션에서와 같이 그 페이지 또는 레이아웃 영역의 크기가 크게 변경될 수 있는 애플리케이션들에서 가장 유용하다. 가변 데이터 프린팅 애플리케이션들에서는, 대개 레이아웃의 품질이 중요하고, 아이템들이 다음 행 또는 열로 랩되는 경우 레이아웃의 외관이 좋지 않으므로, 가변 데이터 프린팅 애플리케이션들에 대해서는 스택킹이 덜 유용하다.

계층적 레이아웃 모델에서, 아이템들의 그룹을 사용하여 복잡한 레이아웃들을 정의할 수 있다. 그룹은 다른 아이템들을 포함하는 직사각형 레이아웃 아이템이고, 그 자체가 레이아웃이다. 그룹 내의 아이템들은 그룹의 직사각형 경계 내부에 레이아웃된다.

레이아웃 내의 아이템들의 위치는 정렬 옵션들(alignment options)에 의해 제어될 수 있다. 예를 들면, 수직 스택 내의 아이템은, 수직 스택을 포함하는 직사각형에서 가능한한 좌측으로 치우칠 수 있도록 좌측 정렬될 수 있다. 정렬은 그룹과 연관될 수 있으므로 그룹 내의 모든 아이템들이 동일한 정렬을 가질 수도 있고, 또는 그룹 내의 각각의 아이템이 대응하는 연관 정렬 옵션들을 가질 수 있다. 통상적으로, 중앙, 좌측 정렬 및 우측 정렬을 포함하는 여러가지 정렬 옵션들이 지원된다.

또한, 아이템들의 위치는 개별 아이템들과 연관된 여백, 또는 개별 아이템을 포함하는 레이아웃 또는 그룹과 연관된 마진들(margins)에 의해 제어될 수 있다. 마진들은 두 개의 인접한 아이템들 사이에서 허용되는 최소값을 정의한다.

제한 기반 모델들(Constraint Based Models)

현재의 계층적 레이아웃 모델들은 이전에 개발된, 보다 일반적인 제한-기반 모델들보다 이해하기가 좀 더 간단하다. 제한-기반 모델들은 캐드(CAD) 애플리케이션들, 사용자 인터페이스 구성 및 윈도우 관리를 위해 사용되어 왔다. 사용자 인터페이스 구성 애플리케이션들에서, 레이아웃될 개체들은 버튼들 및 입력 필드들과 같은 위젯(widget)들이다. 윈도우 관리 애플리케이션들에서, 레이아웃될 개체들은 컴퓨터 운영 체제를 위한 사용자 인터페이스의 일부로서 데이터의 뷰(view)를 나타내는데 사용되는 윈도우들이다.

사용자 인터페이스 구성 애플리케이션들에서, 그래픽 사용자 인터페이스(GUI)가 상이한 폰트 및 텍스트 크기들의 사용과 같은 변화 및 상이한 스크린 해상도들에 적응하도록 하기 위하여, 동적인 레이아웃 제한들이 사용된다. 동적인 레이아웃은 다수의 플랫폼들에 대해 사용자 인터페이스를 구축하는 프로세스를 간소화한다.

윈도우 관리 애플리케이션들에서, 사용자는, 윈도우가 추가 또는 제거됨에 따라 혹은 윈도우가 크기 또는 위치를 변경함에 따라 유지되는 윈도우들 간의 관계를 생성하기 위해 동적인 제한들을 적용할 수 있다.

CAD 애플리케이션들은 매우 일반적인 기하학적 제한들을 포함하며, 그 레이아웃 모델들은 사용하기에 보다 더 복잡하고, 보다 더 어려우며, 레이아웃들은 문서 레이아웃 모델들보다 계산하는데 더 많은 시간이 소모된다. 예를 들어, CAD 애플리케이션들에 사용되는 몇가지 제한 솔버(solver)들은 두 개의 라인들이 평행함을 특정하는 것 - 이는 선형 방정식을 사용하여 표현할 수 없음 - 을 지원한다.

사용자 인터페이스(UI) 및 윈도우 관리 애플리케이션들에 사용된 제한 기반 레이아웃 모델들은, 레이아웃이 그래픽 편집 동작동안 초당 여러회 업데이트 될 필요가 있다면, 상호적으로 사용되도록 충분히 빠를 필요가 종종 있으며, 따라서 레이아웃 방법도 그러한 동작을 지원하도록 충분히 빠를 필요가 있다. 이러한 애플리케이션들은 통상적으로 레이아웃 계산들을 수행하기 위한 전용 제한 솔버를 사용한다. 이러한 애플리케이션들에 사용된 제한 솔버들은 통상적으로, 선형 방정식들(즉, 선형 등식들), 또는 선형 부등식들, 또는 양자 모두로서 표현되는 선형 제한들을 지원한다. 통상적으로, 이러한 제한 솔버들은 또한 솔루션 품질을 정의하는 목적 함수(objective function)를 갖는다. 제한 솔버의 목적은 어떠한 제한도 위반하지 않고도 목적 함수의 값을 최소화(또는 최대화)하는 것이다. 목적 함수 역시 선형이면, 이러한 타입의 문제는 선형 프로그램이라고 불린다. 이러한 타입의 문제들은 이해가 쉬운 최적화 문제들이며, 그들을 해결하기 위한 공지된 알고리즘들이 존재한다. 통상적으로 단순 알고리즘(simplex algorithm)이라고 불리는 알고리즘이 이러한 타입의 문제들을 해결하는데 사용된다.

일련의 제한 및 목적 함수에 의해 정의된 레이아웃 문제가 주어진 경우에, 그 문제는 과잉-제한(over-constrained)되어 어떠한 해결책도 존재하지 않을 수도 있다. 또는 그 문제가 부족-제한(under-constrained)이어서, 많은 해결책들이 존재하거나 한가지 해결책이 존재할 수도 있다. 레이아웃을 계산할 수 있으려면, 레이아웃 모델은 레이아웃들이 정확히 한가지 해결책만을 가진다는 점을 보증할 필요가 있다. 이를 해결하기 위한 한가지 방법은 해결책이 존재하지 않거나, 너무 많은 해결책들이 존재한다는 사실을 사용자에게 보고하고, 사용자가 그 문제를 수정하도록 하는 것이다. 일반적으로, 문제가 과잉-제한되거나 부족-제한된 이유에 대해 사용자에게 의미있는 정보를 제시하는 것이 쉬운 일이 아니므로, 이러한 방법은 수용가능한 해결책이 아니다. 시스템이 과잉-제한된다면, 많은 제한들이 동시에 서로 양립하지 못할 수 있다. 시스템이 부족-제한된 경우, 사용자가 양립할 수 없는 제한들을 추가하는 것을 방지할 수 있는지 그 방법이 분명하지 않을 수도 있다. 제한들이 그래픽 인터페이스를 사용하여 편집되는 경우 이러한 점이 특히 문제가 된다.

해결책이 존재하지 않는 경우를 피하기 위한 한가지 공지된 방법은, 제한 계층(constraint hierarchy)이라 불리는 형태로 제한들의 우선순위를 매기는 것이다. 제한들에 어떠한 해결책도 존재하지 않는다면, 가능한 해결책을 발견할 때까지 우선순위의 역순으로 제한들이 무시된다.

부족-제한된 문제들을 피하기 위한 공지된 방법은 비-선형 목적 함수를 이용하는 것이다. 적절한 엄격 볼록 함수(strictly convex function)를 사용하면, 그 문제는 항상 유일한 해결책을 가질 것이다. 통상적으로, 선형 제한들 및 2차 목적 함수(quadratic objectvie function)를 갖는 최적화 문제들을 해결하기 위한 공지된 기술들이 존재하므로, 2차 목적 함수가 사용된다. 이러한 함수들 중 가장 간단한 것은 단순 알고리즘의 변형물들이다.

가변 데이터 문서들의 생성에 있어서의 또 다른 문제는, 생성되는 문서들을 미리보는(previewing) 것이다. 워드 프로세서들 및 데스크탑 출판 애플리케이션은 종종 프린트 전에 사용자가 작업을 전체적으로 평가하는 것을 돕기 위해 "프린트 미리보기"를 사용한다. 많은 수의 문서들을 미리보기해야 할 필요가 있는 경우, 이러한 기능은 어려운 일이 될 수 있다.

발명이 이루고자 하는 기술적 과제

본 발명의 목적은 가변 문서 프린팅과 연관된 하나 이상의 문제들을 실질적으로 극복하거나 적어도 개선하는 것이다.

발명의 구성 및 작용

본 발명의 일 태양에 따르면, 템플릿에 기초하여 가변 데이터 문서를 위한 레이아웃을 생성하는 방법이 개시되며, 그 방법은:

상기 레이아웃을 형성하기 위해 상기 템플릿 내에 적어도 하나의 콘테이너를 세팅하는 단계;

상기 콘테이너의 하나 이상의 특징 중에서 선택된 각각의 특징에 대해 적어도 하나의 제한을 설정하는 단계;

그래픽 사용자 인터페이스(GUI:graphical user interface)를 이용하여, 상기 설정 단계에서 설정된 상기 제한을 상기 세팅 단계에서 세팅된 상기 콘테이너에 시각적으로 디스플레이하는 단계; 및

콘텐츠를 복수의 콘테이너들에 배치함으로써 상기 문서를 생성하도록 상기 레이아웃을 변경하는 단계를 포함하고,

상기 적어도 하나의 콘테이너의 적어도 하나의 차원(dimension)과 상기 레이아웃 내의 상기 적어도 하나의 콘테이너의 위치 중 적어도 하나는 상기 레이아웃 내의 각각의 제한이 만족되는 것을 조건으로 상기 배치된 콘텐츠의 속성에 기초하여 변경하는 것을 특징으로 한다.

또한, 상기 방법들을 수행하는 장치들 및 컴퓨터 프로그램들을 포함하는 본 발명의 다른 태양들이 개시된다.

1. 개요

소프트웨어 애플리케이션으로서 바람직하게 구현되는 가변 데이터 문서 생성 및 프린팅 시스템이 개시된다. 애플리케이션은 콘텐츠는 다르지만 유사한 형태들을 갖는 복수의 문서들을 생성한다. 이는 문서 템플릿의 생성 및 편집과, 가변 데이터를 갖는 문서 템플릿 상의 영역들의 연관에 의해 달성된다.

문서 템플릿은 복수의 콘테이너들을 포함할 수 있으며, 각각은 텍스트 또는 이미지 데이터와 같은 콘텐츠를 보유하도록 구성된다. 콘테이너들은 위치 및 크기가 고정될 수도 있고, 또는 사용자 지정 규칙에 따라 문서마다 차원 또는 위치가 달라질 수 있다. 이러한 콘테이너들의 콘텐츠는 정적이거나 또는 가변적일(즉, 데이터베이스와 같은 몇가지 데이터 소스에 의존할) 수 있다.

사용자들은 데이터를 획득하기 위한 데이터 소스들의 여러가지 타입들을 지정할 수 있다. 데이터 소스는 복수의 데이터 레코드들(data records)을 포함한다. 애플리케이션은 문서 템플릿 내의 콘테이너들과 데이터를 연관시키기 위한 메카니즘을 제공한다.

이어서, 문서 템플릿 내의 데이터 소스와 콘테이너들 간의 연관성에 기초하여 문서 템플릿 및 데이터 소스로부터의 데이터를 "통합(merge)"함으로써 많은 문서들이 생성된다. 통상적으로, 데이터 소스 내의 각각의 레코드 또는 레코드들의 그룹에 대해 하나의 문서가 생성된다. 그 후, 문서들은 디스크에 저장되거나 필요에 따라 프린트될 수 있다.

이러한 과정들은 종래 기술에 비해 몇가지 독립적인 이점을 제공한다. 예를 들면, 종래 기술과는 달리, 문서 템플릿을 편집하는 경우에 사용자는 통합된 문서들 중 하나를 항상 보도록 선택할 수 있다. 이는 문서 템플릿과 데이터가 동적으로 통합되기 때문이다. 결과적으로, (문서 템플릿 대신에) 문서가 어떻게 보일 것인지를 알기 위해 특정 미리보기 영역으로 갈 필요가 있는 종래 기술과는 달리, 적어도 그 결과 문서들 중 하나는 그 템플릿을 편집하는 동안 어떻게 보일 것인지를 알 수 있다.

(콘테이너들이 이동하거나 크기를 변경할 수 있는 방법을 결정하는) 콘테이너에 대한 제한들이 동일한 위치에 도시되며, 종래 기술과는 달리 별도의 스크린 또는 영역으로 전환할 필요없이 콘테이너들 상의 여러 위치들을 클릭함으로써 제한들이 그 자리에서 편집될 수 있다. 이는 종래 기술과 비교하여 문서 템플릿의 생성을 간소화한다.

2. 구현예들의 개요

기본적인 구현예는 적어도 사용자 인터페이스 및 레이아웃 엔진을 포함하는 가변 데이터 문서 생성 및 프린팅 애플리케이션이다. 한가지 특정 구현예는 프린터와 함께 퍼스널 컴퓨터 상에서 실행 가능한 소프트웨어이다. 또 다른 구현예에서, 애플리케이션은 가변 데이터 문서 세트들의 프린팅을 지원하는 프린터 또는 프린트 컨트롤러에 포함된 처리기 상에서 실행 가능한 소프트웨어로서 통합된다. 또 다른 구현예에서, 애플리케이션은 뷰어에 대해 맞춤제작된 문서들을 제공할 수 있는, 웹 서버에서 실행가능한 소프트웨어로서 통합된다. 이러한 구현예들은 또한 적절한 하드웨어 장치에서 실행되는 경우에 이러한 소프트웨어를 포함한다.

가변 데이터 프린팅은 관련 문서들의 세트들을 프린팅하는 것을 의미한다. 비-가변(non-variable) 데이터 프린팅에서, 문서들은 개별적으로 프린팅되고 각각의 문서에 대한 데이터는 통상적으로 프린터에 개별적으로 송신된다. 가변 데이터 프린팅에서는, 다수의 관련 문서들을 포함하는 프린트 작업(job)이 프린터로 송신된다. 통상적으로, 가변 데이터 프린트 작업에서의 문서들은 다수의 문서들에서 발생하는 공유 요소들을 포함한다. 이러한 데이터는 통상적으로 각각의 문서에 대해 반복되는 대신에 작업당 단지 한번만 프린터로 송신된다. 가변 데이터 프린트 작업이 프린트될 때, 공유 데이터가 각각의 문서에 삽입된다. 이러한 작업은, 페이지들이 프린트되는 시점에 문서의 페이지들의 레이아웃을 계산해야 할 필요가 있다. 이러한 경우에서, 레이아웃 방법은 프린터 또는 프린트 서버 내의 소프트웨어의 일부일 수 있다. 보다 통상적으로는, 각각의 문서의 레이아웃은 클라이언트 장치 상에서 계산되고, 레이아웃은 프린트 작업의 일부로서 프린터로 송신되는데, 이 경우 레이아웃 방법이 클라이언트 장치 상에서 수행된다.

3. 시스템 설명

도 1a는 가변 데이터 문서들을 프린트하기 위한 시스템(100)을 도시한다. 본 명세서에 기재된 방법들은 도 1b에서 상세하게 서술된 범용 컴퓨터 모듈(101) 내에서 실시될 수 있으며, 기재된 프로세스들은, 시스템(100)을 통해 동작 가능하고 컴퓨터 모듈(101) 내에서 실행되는 레이아웃 편집 애플리케이션 프로그램(121)과 같은 소프트웨어에 의해 전체적으로 또는 부분적으로 구현될 수 있다. 특히, 레이아웃 편집 및 결과적인 프린팅 단계들은 컴퓨터(101)에 의해 수행되는 소프트웨어 내의 명령들에 의해 구현될 수 있다. 소프트웨어는 예를 들어, 이하에 기재된 저장 장치들을 포함하는 컴퓨터 판독가능 매체에 저장될 수 있다. 소프트웨어는 컴퓨터 판독가능 매체로부터 컴퓨터 내로 로딩되어, 컴퓨터(101)에 의해 실행된다. 그러한 소프트웨어 또는 컴퓨터 프로그램을 수록한 컴퓨터 판독가능 매체는 컴퓨터 프로그램 제품이다. 컴퓨터에서 컴퓨터 프로그램 제품을 사용함으로써, 바람직하게는 문서 레이아웃 편집 및 가변 문서 프린팅을 위한 유익한 장치를 얻게 된다.

컴퓨터 모듈(101)은 키보드(132)와 같은 입력 장치들, 마우스(133)와 같은 포인팅 장치, 및 디스플레이 장치(144)를 포함하는 출력 장치들 그리고 선택적으로는 로컬 프린터(145)에 결합된다. 입/출력 인터페이스(138)는, 컴퓨터 모듈(101)을 네트워크 접속(107)을 통해 시스템(100) 상의 다른 컴퓨팅 장치들로 결합시킨다. 네트워크 접속(107)은 통상적으로 근거리 네트워크(LAN) 또는 원거리 네트워크(WAN)이다.

컴퓨터 모듈(101)은 통상적으로 적어도 하나의 처리기 유닛(135), 예컨대 반도체 RAM 및 ROM로부터 형성된 메모리 유닛(136), 비디오 인터페이스(137)를 포함하는 입/출력(I/O) 인터페이스들, 키보드(132) 및 마우스(133)를 위한 입/출력 인터페이스(143)를을 포함한다. 저장 장치(139)가 제공되며, 이는 통상적으로 하드 디스크 드라이브(140) 및 플로피 디스크 드라이브(141)를 포함한다. 자기 테이프 드라이브(도시되지 않음)가 또한 사용될 수 있다. CD-ROM 드라이브(142)가 데이터의 비휘발성 소스로서 통상적으로 제공된다. 컴퓨터 모듈(101)은 GNU/Linux 또는 Microsoft Windows와 같은 운영체제를 이용하고, 컴퓨터 모듈(101)의 구성요소들(135 내지 143)은 통상적으로 운영체제에 따라 상호접속된 버스(134)를 통해 통신하며, 이로 인해 컴퓨터 시스템은 당업자에게 공지된 종래의 모드로 동작한다. 기술된 장치들이 실시될 수 있는 컴퓨터들의 예는 IBM-PC 및 호환가능한 썬 스파크스테이션(Sun Sparcstations) 또는 그로부터 진화된 유사한 컴퓨터 시스템들을 포함한다.

통상적으로 레이아웃 편집 애플리케이션 프로그램(121)은 하드 디스크 드라이브(140) 상에 존재하고 프로세서(135)에 의한 실행을 통해 판독 및 제어된다. 프로그램(121) 및 네트워크(107)로부터 인출된 임의의 데이터의 중간 저장이 하드 디스크 드라이브(140)와 함께 반도체 메모리(136)를 사용하여 수행될 수 있다. 일부 경우들에서, 애플리케이션 프로그램(121)은 CD-ROM 또는 플로피 디스크 상에 인코딩되고 대응하는 드라이브(142 또는 141)를 통해 판독되어 사용자에게 제공되거나, 또는 대안으로 네트워크 접속(107)으로부터의 사용자에 의해 판독될 수 있다. 더욱이, 소프트웨어는 또한 자기 테이프, ROM 또는 집적 회로, 광자기 디스크, 컴퓨터 모듈(101)과 또다른 디바이스 간의 무선 또는 적외선 전송 채널, PCMCIA 카드와 같은 컴퓨터 판독가능 카드, 및 웹사이트들 상에 기록된 이메일 전송들 및 정보를 포함하는 인터넷 및 인트라넷 등을 포함하는, 다른 적절한 크기의 컴퓨터 판독가능 매체로부터 컴퓨터 모듈(101)로 로딩될 수 있다. 앞서 기술한 것은 단지 관련 컴퓨터 판독가능 매체의 예이다. 다른 컴퓨터 판독가능 매체가 사용될 수 있다.

레이아웃 편집으로 이름 붙여진 애플리케이션(121)은 가변 데이터 프린팅(Variable Data Printing; VDP)을 수행하도록 동작하고 두 개의 소프트웨어 구성요소들을 포함한다. 이러한 구성요소들의 첫 번째는 레이아웃 엔진(105)이며, 이는 직사각형 영역 내에서, 직사각형 및 라인들의 위치들을 주어진 제한들 및 크기하에서 계산하기 위한 소프트웨어 구성요소이다. 두번째로, 사용자 인터페이스(103)컴포넌트는 사용자가 문서 템플릿을 구성하고, 데이터 소스들과 문서 템플릿 내의 영역들을 연관시키도록 하기 위한 매카니즘을 제공한다. 사용자 인터페이스(103) 및 레이아웃 엔진(105)은 통신 채널(123)을 통해 통신한다. 문서 생성을 위한 데이터 소스는 통상적으로 데이터베이스 서버(117) 상에 수용되고, 일반적으로 데이터베이스 애플리케이션을 실행하는 또 다른 컴퓨터에 의해 형성되는 데이터베이스(119)이다. 호스트 컴퓨터(101)는 네트워크 접속(107)을 통해 데이터베이스 서버(117)와 통신한다. 가변 데이터 프린팅 애플리케이션(121)은, 호스트 컴퓨터(101) 또는 일반적으로 다른 컴퓨터로 이루어지는 파일 서버(115)에 저장될 수 있는 문서 템플릿들을 생성한다. 또한, 가변 데이터 프린팅 애플리케이션(121)은 데이터와 통합된 문서 템플릿에 의해 형성된 문서들을 생성한다. 이러한 문서들은 호스트 컴퓨터(101) 상의 로컬 파일 시스템에 저장되거나, 파일 서버(115)에 저장되거나, 프린팅을 위해 직접 프린트 서버(109) 또는 프린터(113)로 송신될 수 있다. 프린트 서버(109)는 직접적으로 네트워킹될 수 없는 프린터들에 네트워크 기능을 제공하는 컴퓨터이다. 프린트 서버(109) 및 프린터(113)는 전형적인 통신 채널(111)을 통해 접속된다.

도 2는, 별도의 버전(seperate version; 225)의 레이아웃 엔진(105)을 포함하는 엔진 서버(227)가 추가되었다는 점을 제외하고는 도 1과 유사하다. 엔진 서버(227)는 또 다른 전형적인 컴퓨터이다. 파일 서버(115) 상에 저장된 문서 템플릿들은, 프린팅 또는 다른 목적을 위해 데이터베이스(119) 내에 저장된 문서들을 생성하기 위한 데이터와 레이아웃 엔진(225)에 의해 결합될 수 있다. 이러한 동작은 사용자 인터페이스(103)를 통해 요청될 수 있거나 단지 프린트될 특정 레코드를 요청할 수 있다.

4. 메인 윈도우(Main Window)

도 3을 참조하면, 사용자 인터페이스(103)는 동작시에 비디오 디스플레이(144) 상에 디스플레이되는 애플리케이션 윈도우(301)에 의해 형성된 그래픽 사용자 인터페이스를 포함한다. 윈도우(301)는, 일부 구현에에서 스크린 상의 다양한 위치로 옮겨지거나 분리될 수 있는 툴바 영역(303), 메뉴 바(302), 작업 영역(306), 선택적 플로팅 팔레트(optional floating palette; 311) 및 커서/포인터 디바이스(313)를 특징으로 하는데, 커서/포인터 디바이스(313)의 위치는 통상적으로 마우스(133)의 위치 또는 움직임과 연관된다.

메뉴 바(302)는, 활성화된 경우, 본 기술 분야에서 통상적인 메뉴 옵션들의 계층으로 확장되는 많은 메뉴 아이템들(304)을 갖는다.

툴바(303)는, 각각이 애플리케이션의 특정 모드에 의존하여 숨겨지거나 보여지는 많은 툴 버튼들 또는 위젯들(305)을 갖는다.

선택적 눈금자들(308)이 포인터, 페이지들, 라인들, 마진 가이드들, 콘테이너들 또는 작업 영역 내의 다른 개체들의 위치를 표시하는데 사용될 수 있다. 눈금자들(308)은 예를 들어, 인치, 밀리미터 또는 픽셀들과 같이, 사용되는 단위들의 수치적인 표시를 보여줄 수 있다.

가변 데이터 라이브러리와 같은 부가적인 기능들에 접근하기 위해 플로팅 팔레트(311)가 사용될 수 있다. 팔레트(311)는 이동, 리사이징 및 종료의 기능을 수행하도록 하는 자신의 윈도우 제어들(312)을 갖는다. 팔레트(311)는 선택적으로 항상 작업 영역의 앞에 존재할 수 있으며, 또는 다른 대상들 뒤에 숨겨질 수도 있다. 팔레트(311)는 애플리케이션 윈도우(301)의 범위들 내에만 나타나도록 제한되거나, 또는 애플리케이션 윈도우(301)의 외부에 부분적으로 또는 전체적으로 나타나도록 허용될 수 있다.

도 4를 참조하면, 적어도 다음의 사용자 선택가능 아이콘 모양 "버튼들"을 갖는 툴바 영역(303)이 도시되어 있다:

● 선택 툴 버튼(403): 콘테이너 에지들을 선택, 이동, 스케일링, 리사이징 및 잠금/잠금해제하는데 사용됨. 또한, 콘테이너들 주위로 선택 박스를 드래그하거나 CTRL 키를 누른 채로 콘테이너들을 선택함으로써, 콘테이너들은 복수로 선택될 수 있다.

● 이미지 콘테이너 툴 버튼(405): 정적 또는 가변 이미지들을 포함하기 위한 콘테이너들을 생성하는데 사용됨.

● 텍스트 콘테이너 툴 버튼(404): 정적 또는 가변 텍스트를 포함하기 위한 콘테이너들을 생성하는데 사용됨.

● 스트럿 툴 버튼(406): 콘테이너들 간의 거리를 제어하는데 사용됨.

이러한 버튼들은 본 기술분야에서 공지되어 있는 바와 같이, 콘텍스트 변동 툴 팁들(context sensitive tool tips)로 구현될 수 있다.

5. 문서 템플릿

문서 템플릿의 디자인을 확인 및 편집하기 위하여 작업 영역(306)이 사용된다. 이로 인해 사용자는 준비 중의 문서들의 프린트된 외관을 디자인하고, 각각의 통합된 문서가 문서 템플릿과 통합된 가변 데이터의 양 및 크기에 기초하여 어떻게 변화하는지를 이해할 수 있다.

외부 데이터 소스가 템플릿으로 링크되면, 가변 텍스트 및 이미지들은 자신들의 콘테이너에 디스플레이 되어, 사용자들이 그들이 작업한 대로 현재 문서를 미리볼 수 있다.

사용자가 커서를 롤오버하거나 콘테이너를 선택할 때마다 문서의 구조 및 가변 데이터 콘테이너들의 동작을 기술하는 시각적인 단서들이 디스플레이된다.

작업 영역(306)은 스크롤 바(307), 선택적 눈금자들(308) 및 문서 템플릿(309)을 특징으로 한다. 문서 템플릿(309)는 복수의 페이지들을 보여줄 수 있다.

당해 기술 분야에 공지된 바와 같이, 주어진 문서 템플릿에 대한 페이지 크기는 사용자에 의해 지정된다. 각 문서 내의 실제 페이지 수는 가변 데이터에 따라 변경될 수 있다. 모든 가변 데이터가 한 페이지에 들어가지 않으면, 데이터를 디스플레이하기 위해 부가적인 페이지들이 자동으로 생성될 수 있다.

페이지 상의 프린트 가능한 개체들의 최대 범위를 나타내는 선택적인 페이지 마진(310)은 각각의 페이지 경계 내에 존재한다.

또한 도 4에 도시된 것은 문서 템플릿(309)의 페이지 상에 나타날 수 있는 복수의 개체들의 예이며, 그들은: 선택적 앵커 아이콘(409), 비고정된 에지들(410), 스트럿(412) 및 슬라이더들(413)을 특징으로 하는 다수의 콘테이너들(407 및 408)이다.

6. 콘테이너들

콘테이너는, 텍스트, 이미지들 및 다른 콘테이너들 또는 개체들과 같은 정적 또는 동적 콘텐츠가 배치될 수 있는 문서 템플릿 내의 공간이다. 콘테이너들은 포인팅 디바이스(313)를 사용하여 사용자 인터페이스에 나타난 콘테이너를 조작함으로써, 콘테이너를 이동시키고, 크기를 변경하며, 모양을 바꿀 수 있고, 마우스(133)를 통해 제어할 수 있다.

보다 자세히 설명하면, 콘테이너는 세팅들, 시각적 외관들 및 상호작용 및 편집 동작들의 집합을 갖는다. 다음은 콘테이너에 대한 정의의 모든 부분이다:

● 콘테이너는 정적 및/또는 동적 콘텐츠를 가질 수 있다. 동적 콘텐츠는 데이터 소스로부터 유래하고, 다른 문서들에 대해 상이할 수 있다는 의미에서 동적이다. 동적 콘텐츠는 움직이는 개체(animated) 또는 시간에 따라 변화하는 콘텐츠를 포함하도록 의도되지는 않았는데, 이러한 것들은 프린팅에 적합하지 않기 때문이다. 마찬가지로, 동적 콘텐츠의 동작으로 인해 정적 콘텐츠가 각각의 문서에 대해 다르게 배치될 수 있을지라도, 정적 콘텐츠는 이러한 콘테이너를 사용하여 생성된 모든 문서들에서 동일하게 나타날 것이다.

● 콘테이너는 배경색, 경계 그리고 폰트 및 스타일과 같은 텍스트 세팅들과 같은 외관상의 특징들을 가질 수 있는데, 이들은 콘테이너의 콘텐츠에 적용된다.

● 문서를 생성하는 경우, 콘테이너는 데이터 소스로부터의 데이터와 통합될 수 있다. 임의의 정적 콘텐츠와 같은 외관상의 특징들은 통상적으로 프린트된 출력에서 시각화될 수 있다. 동적 콘텐츠는 데이터 소스로부터의 특정 데이터의 외관을 초래할 것이다. 이러한 콘테이너의 표현은 예를 들어, 프린트 또는 스크린(144) 상에 디스플레이되거나, 양자 모두가 될 수 있다.

● 또한 콘테이너는, 예를 들어, 콘테이너의 세팅들을 편집하고 확인하기 위한 상호적인 그래픽 사용자 인터페이스와 같은 사용자 인터페이스를 가질 수 있다. 인터페이스 요소들은 통상적으로 스크린(144) 상에는 나타나지만 프린트된 문서들에는 나타나지 않는다. 사용자 인터페이스(103)는 배경색 또는 폰트와 같은 콘테이너의 외관상의 특징들의 일부를 디스플레이할 수 있고, 또한 콘테이너의 세팅들의 편집 및 확인을 허용하는 특징들을 추가할 수도 있다. 전용 사용자 인터페이스 특징들의 예들은, 콘테이너가 데이터 소스로부터의 데이터와 통합된 경우 그 콘테이너의 동작을 나타내기 위해, 콘테이너의 크기 또는 위치를 대화식으로 디스플레이하고 변경하기 위한 경계들 또는 코너 아이콘들, 또는 오버레이된 숫자들, 선들, 아이콘들 또는 텍스트이다.

본 개시의 일 태양은 일련의 새로운 직접 조작 기술 및 콘테이너의 그래픽 사용자 인터페이스 구성요소를 포함하는 디스플레이 방법이다.

6.1 콘테이너 제한들

본 개시에 따르면, 콘테이너는 연관된 콘텐츠가 각각의 문서에서 어떻게 디스플레이될 수 있는지를 제어하는 몇가지 제한들을 가질 수 있다. 콘테이너와 정적 및 동적 콘텐츠를 연관시키는 수단과 함께, 이러한 제한들은 사용자가 단일의 문서 템플릿으로부터 다수의 문서들의 생성을 제어하는 주요 방법이다. 제한의 예는 "이러한 콘테이너의 콘텐츠는 최대 4인치 크기가 될 수 있다"이다. 또 다른 제한은 "이러한 콘테이너의 콘텐츠의 좌측 에지는 각각의 문서에서 동일한 수평 위치에 나타내야 한다"가 될 수 있다. 그래픽 사용자 인터페이스를 사용하여 이러한 제한들을 디스플레이하고 편집하기 위한 일련의 방법들이 본 명세서에서 기술된다.

페이지 상에 일부 정의된 위치를 갖는 이미지와 같은 정적 콘텐츠의 위치들을 지정하는 콘텐츠 위치고정자(placeholder)들은 디지털 프린팅 기술 분야에 잘 알려져 있다. 다음의 논의에서, 콘테이너들은 위치 및 크기를 가질 수 있고, 이들은 본 기술분야에 공지된 것과 유사한 방식으로 디스플레이되고 편집될 수 있음을 가정한다. 대신에, 본 논의는 가변 데이터 프린팅에 특정한 디스플레이 및 편집 방법들에 중점을 둔다.

콘테이너들은, 사용자가 문서들 내의 콘텐츠의 크기 및 위치를 지정하도록 허용한다. 단일의 문서 템플릿으로부터 여러 문서들이 생성될 수 있으므로, 콘테이너는 다수의 가능성 및 제한들을 지정하고 디스플레이하기 위한 사용자 인터페이스를 가져야 한다.

콘테이너의 에지들은 연관된 콘텐츠가 문서들에 표시될 가상의 경계를 정의한다. 그러므로, 본 특허 명세서에서는, 콘테이너의 좌측 에지에 대해 논의하는 것은 연관된 콘텐츠가 생성된 임의의 문서들에 디스플레이될 수 있는 최좌측 에지(left-most edge)를 논의하는 것과 동일할 수 있다. 유사하게, 콘테이너의 높이에 대해 논의하는 것은 생성된 임의의 문서들 내의 연관된 콘텐츠의 높이에 대한 제한에 대해 논의하는 것으로서 이해될 수 있다. 본 특허 명세서가 사용자 인터페이스(103)를 참조하여 콘테이너의 에지 또는 크기를 논의함에 있어서, 이러한 구별이 명확해질 것이다.

다음의 논의에서, 용어 '고정된'은 콘텐츠의 외관을 제어하는데 사용된 일부 값이 모든 문서에서 동일함을 정의한다.

● 콘테이너의 폭이 고정되면, 이는 연관된 콘텐츠에 허용된 폭이 모든 문서에서 동일함을 의미한다.

● 콘테이너의 높이가 고정되면, 이는 연관된 콘텐츠에서 허용된 높이가 모든 문서에서 동일함을 의미한다.

● 거리 제한이 고정되면, 지정된 거리는 모든 문서들에 대해 일정하다.

● 콘테이너의 좌 또는 우측 에지가 고정되면, 이는 페이지에 대한 해당 에지의 수평 위치가 모든 문서에 대해 동일하지만, 콘테이너의 높이 또는 수직 위치는 변경될 수 있음을 의미한다. 예를 들어, 콘테이너의 좌측 에지가 고정되면, 연관된 콘텐츠는 하나의 문서에서 페이지의 상부 근처, 및 다른 한편으로는 페이지의 하부 근처에 나타날 수 있지만, 좌측 에지는 모든 경우에서 동일한 수평 위치를 가질 것이다.

● 콘테이너의 상부 또는 하부 에지가 고정되면, 이는 페이지에 대한 에지의 수직 위치가 모든 문서에 대해 동일하지만, 콘테이너의 폭 또는 수평 위치는 변경될 수 있음을 의미한다.

● 콘테이너의 수직축은 콘테이너의 좌 및 우측 에지들에 평행하고 그들 사이의 중간 정도에 위치된 가상의 수직선이다. 콘테이너의 수직축이 고정되면, 콘테이너의 좌 및 우측 에지들의 수평 위치의 평균은 모든 문서에 대해 동일할 것이다. 이러한 제한에 따라, 콘테이너의 폭이 변경될 수 있으므로, 좌 및 우측 에지들 양자 모두는 상이한 문서들에서 수직축에서 더 멀거나 더 가까울 수 있으나, 수직 축은 모든 문서들에 대해 동일한 수평 위치에 존재한다. 콘테이너의 높이 및 수직 위치는 이러한 제한에 의해 영향받지 않는다.

● 유사하게, 수평축이 고정되면, 이는 콘테이너의 상부 및 하부 에지들의 수직 위치를 제한하지만, 높이는 이러한 제한에 의해 영향 받지 않는다.

● 수평 및 수직 축들 양자 모두가 고정되면. 이는 콘테이너의 중심점이 고정됨을 의미하지만, 폭 및 높이는 이러한 제한들에 의해 영향 받지 않는다.

● 콘테이너의 코너, 콘테이너의 에지의 중점, 또는 콘테이너의 중심점이 고정되면, 이는 해당 점이 모든 문서들 내의 동일한 위치 및 콘테이너에 대해 동일한 위치에 나타남을 의미한다. 예를 들어, 콘테이너의 상부-좌측 코너가 고정되면, 이는 콘텐츠들이 배치될 수 있는 상부-좌측 점이 모든 문서들에 대해 동일함을 의미한다.

● 수직 에지 또는 축은 페이지의 좌측 에지, 또는 우측 또는 좌측 페이지 가장자리, 또는 우측 페이지 가장자리, 또는 몇가지 다른 수평 위치에 대해 고정될 수 있다. 유사하게, 수평 에지 또는 축은 페이지의 상부 또는 하부 에지들, 일부 다른 수직 위치에 대해 고정될 수 있다. '고정된'이라는 용어에 대한 이러한 세부적인 정의는, 페이지 크기가 문서들 간에 변경되는 경우에만 의미가 있는데, 그 이유는 페이지 크기가 모든 문서에 대해 동일하다면 이러한 가능성들이 생성된 문서들에서 아무런 차이도 나타내지 못하기 때문이다.

'고정된'의 반대는 에지, 축, 코너, 중점 또는 거리 제한이 문서들 간에서 변경할 수 있음을 의미하는 '비고정된'이지만, 문서들의 특정 세트에서 반드시 그러할 필요는 없다. 예를 들어, 실질적으로 에지의 위치가 변경되는 것을 방지하는 다른 외부 제한들이 있을 수 있지만, 외부 제한들이 적용되지 않았다면 에지 위치가 변경될 수 있다는 사실은 에지가 여전히 비고정으로 라벨링되었음을 의미한다.

6.2 콘테이너 상태들

콘테이너의 '에지 상태'는 앞서 기재된 바와 같이: 좌측 에지, 우측 에지, 수직축들, 상부 에지, 하부 에지, 수평축, 폭 및 높이의 '고정'되거나 또는 '비고정'될 수 있는 8가지 속성들의 세트로서 정의된다. 에지들은 최소 또는 최대 콘테이너 크기 세팅들 또는 다른 콘테이너들의 위치와 같은 다른 제한들에 의해 영향을 받을 수 있기 때문에, 에지 상태가 에지의 위치를 완전히 제한하지 못할 수도 있다. 그러므로, '에지 상태'는 콘테이너가 가질 수 있는 완전한 세트의 상태들의 단지 일부만을 의미한다.

콘테이너에 대한 그래픽 사용자 인터페이스는, 부분적으로, 그래픽 표현들로 에지 상태들을 매핑하는 것으로부터, 그리고 에지 상태에 대한 조정들로 에지들 및 콘테이너 표현들의 직접적인 조작을 매핑하는 것으로부터 얻어지기 때문에, 에지 상태는 대부분의 구현들에서 중요하다.

바람직하게는, 콘테이너의 완전 상태는 에지 상태 이상을 포함한다. 콘테이너들은 폭 및 높이에 대한 최소 및 최대 세팅들을 가질 수 있으며, 이는 에지 위치들을 더욱 제어한다. 내부 가장자리들은 연관된 콘텐츠가 디스플레이될 수 있는 위치를 제어하며, 이는 차례로 콘테이너 에지들이 배치될 수 있는 위치에 영향을 미칠 수 있다. 이미지 크로핑(image cropping), 스케일링 등 뿐만 아니라 텍스트 폰트, 스타일, 크기 및 정렬 세팅들은 콘테이의 상태의 일부가 될 수 있으며, 콘텐츠 위치, 크기 및 외관에 영향을 미칠 수 있다. 배경 및 경계색과 같은 외관상의 세팅들, 경계선 두께들, 스타일들 등이 또한 각각의 콘테이너 상태의 일부이다. 이러한 측면들이 문서 내의 콘텐츠의 위치들에 영향을 미치지 않을 수 있지만, 외관에는 영향을 미칠 것이다.

일부 구현예에서, 에지 상태는 고정되거나 비고정될 수 있는 상술한 8가지 속성들을 포함한다. 페이지 크기들은 변경될 수 없으므로, '고정된'이라는 용어는 여기서 어떠한 추가적인 한정도 하지 않는다. 이러한 8가지 속성들로 인해 256개의 가능성에 해당하는, 2^8 가지의 구분된 에지 상태가 생성된다.

이러한 모든 256개의 에지 상태들이 사용될 필요가 있는 것은 아니다. 예를 들어, 콘테이너의 좌측 에지가 고정된 상태를 고려하면, 수직축은 고정되고 우측 에지는 고정되지 않는다. 수직축이 고정되어 있기 때문에, 모든 문서들에 대해, 좌측 에지로부터 수직축까지의 거리는 우측 에지로부터 수직축까지의 거리와 동일해야 한다. 또한 좌측 에지가 고정되어 있기 때문에, 모든 문서들에 대하여 수직축으로부터 좌측 에지까지의 거리가 일정하며, 또한 이는 우측 에지가 고정되어 있음을 암시한다.

일정한 구현예들은 에지 상태들의 부분집합을 모델링하며, 이들의 각각은 유일한 동작을 나타낸다. 상술한 에지 상태는 그 동작이 몇가지 다른 상태들과 동일하기 때문에 모든 구현예에서 사용되지는 않는다. 대안적인 구현예는 유일한 콘테이너 동작들을 모델링하는 대신에, 에지들이 고정되었는지 모델링하고 편집하는 수단으로서, 사용자 인터페이스에 이러한 상태들을 나타내고, 이들을 사용할 수 있다.

한 가지 특정한 구현예는 6개의 고유한 수평 제한 및 6개의 고유한 수직 제한들의 곱으로부터 형성된 36개의 콘테이너 상태들을 사용한다.

콘테이너 상의 수평 제한들은:

1. 좌 및 우측 에지들, 수직 축 및 폭이 비고정되는 제한;

2. 좌측 에지만이 고정되는 제한;

3. 우측 에지만이 고정되는 제한;

4. 좌측 및 우측 에지들 모두가 고정되는 제한;

5. 수직축만이 고정되는 제한; 및

6. 폭은 고정되지만 좌측 및 우측 에지들과 수직축은 비고정되는 제한

이 있다.

콘테이너 상의 수직 제한들은:

1. 상부 및 하부 에지들, 수평축 및 높이가 비고정되는 제한;

2. 상부 에지만이 고정되는 제한;

3. 하부 에지만이 고정되는 제한;

4. 상부 및 하부 에지들 모두가 고정되는 제한;

5. 수평축만이 구정되는 제한;

6. 높이는 고정되지만 상부 및 하부 에지들과 수평축은 비고정되는 제한

이 있다.

'고정된'이라는 용어가 한정자(qualifier)를 갖는다면, 예컨대, 각각의 에지 또는 축이 페이지의 좌측 또는 우측 에지 중 하나에 대해 고정될 수 있다면, 이로 인해 36개가 넘는 고유한 에지 상태들이 생길 것이다.

상기 리스트에서 "좌측 및 우측 에지들 양자 모두 고정되는 제한"으로 이름 붙은 제한은 "좌측 및 우측 에지들, 수직축 및 폭 모두 고정되는 제한", 및 "좌측 에지 및 수직축이 고정되는 제한" 그리고 여러 다른 유사한 제한들과 동일한 동작들을 가짐에 주의해야 한다. 에지 상태들이 보다 적은 '고정된' 제한들을 사용한다면, '고정된'의 다양한 개념들을 표현하기 위해 시각 큐들(visual cues)을 사용하는 구현은 그러한 큐들을 보다 적게 사용할 수 있으므로, 보다 단순하고, 대칭적인 제한이 일반적으로 바람직하며, 결과적으로 이는 그렇지 않은 경우보다 적은 시각적인 혼란만을 야기할 수 있다.

이하에서 기술하는 바와 같이, 본 개시의 일 태양은 콘테이너들을 위한 그래픽 표현들 및 편집 방법과 에지 상태들을 연관시킨다.

6.3 콘테이너 디스플레이 및 편집

6.3.1 새로운 콘테이너를 생성하기 위한 방법

텍스트 콘테이너 및 이미지 콘테이너의 두 가지 종류의 콘테이너가 기술된다. 텍스트 콘테이너는 텍스트 및/또는 포함된 이미지들을 포함한다. 이미지 콘테이너는 오직 이미지들만을 포함한다.

도 4를 참조하면, 마우스(133)를 사용하여 텍스트 콘테이너 툴(404) 또는 이미지 콘테이너 툴(405)을 각각 클릭하고 이어서 템플릿(309) 내의 직사각형을 드래그함으로써, 새로운 텍스트 콘테이너들 및 이미지 콘테이너들이 문서 템플릿(309)에 생성될 수 있다.

대안으로, 적절한 툴(404 및 405)을 활성화한 후에 문서 템플릿(309)에서 클릭함으로써, 콘테이너가 간단히 생성될 수 있다. 디폴트 크기의 콘테이너가 삽입되거나, 새로운 콘테이너의 차원들을 입력하기 위한 대화 박스 또는 다른 프롬프트가 제공된다. 몇가지 콘테이너들은 몇가지 미리 정의되거나 또는 계산된 스키마에 따라 자동으로 생성되고 배치될 수 있다. 다른 대안들이 고안될 수 있다.

6.3.2 콘테이너들을 디스플레이하기 위한 방법

바람직하게는, 앞서 기재된 36개의 에지 상태들 각각이 그래픽 표현으로 매핑된다. 몇몇 에지 상태들이 일정한 환경에서 표현을 공유할 수 있기 때문에, 그래픽 표현들의 수는 36개보다 더 적을 수 있다.

도 5a 내지 도 5d는 콘테이너에 대한 제1 예시적인 에지 규칙들을 도시한다.

애플리케이션(121)은 실선들(항목 503 참조) 또는 점선들(504 참조)로서 에지들을 그리며, 또한 에지 상태를 나타내기 위해 앵커들(에지 상에 또는 에지 근처에 그려지고 506, 507 및 509로 도시된 선들, 모양들 및/또는 아이콘들로 이루어짐), 핸들들(이동되거나 변경되도록 허용하기 위해 에지 상에 또는 에지 근처에 그려진 제어 점들 또는 모양, 502 참조), 슬라이더들(에지의 한쪽 측면 상에 그려진 짧은 평행선들, 도 4 참조, 413으로 라벨링된 항목들), 확대 아이콘들(505 참조), 및 색상들을 특징 짓는다.

도 5a 내지 도 5d의 콘테이너 디스플레이 방법에 대한 규칙들은, 차례로:

1. 각각의 고정된 에지에 대해 에지를 실선으로 그리고;

2. 폭이 고정되면, 좌측 및 우측 에지들을 실선으로 그리며;

3. 높이가 고정되면, 상부 및 하부 에지들을 실선으로 그리고;

4. 축들은 그리지 않으며;

5. 아직 그려지지 않은 모든 에지들은 각각의 에지 근처에 그려진 확대 아이콘으로 점선으로 그려지고;

6. 수직 에지들 및/또는 축들의 각 쌍들에 대해, 양자 모두가 고정되면 그들의 교차점에 앵커를 그리며;

7. 각각의 고정된 에지에 대해, 어떠한 앵커들도 그 에지 상의 어디에도 그려지지 않았다면 에지의 중심에서 슬라이더를 그리고;

8. 수직 에지들 및/또는 축들의 각각의 쌍에 대해, 어떠한 앵커 또는 슬라이더도 그 위치에 그려지지 않았다면 그들의 교차점에 핸들을 그리는 것이다.

규칙들 1, 2 및 3은 라인들이 고정되거나 제한되는 경우 실선으로 그려짐을 보장한다. 규칙 5는 비고정된 에지들이 점선으로 그려짐을 보장한다. 규칙들 6, 7 및 8은 고정된 점들이 앵커들을 디스플레이하고, 몇몇 고정된 에지들이 슬라이더들을 디스플레이하며, 다른 것들이 핸들을 디스플레이함을 보장한다.

상기에서, 에지들은 단지 한번만 그려질 필요가 있으므로, 규칙에 의해 에지가 그려지면 다음의 규칙들은 에지가 다시 그려지도록 하지 않을 것이다. 아이콘들은 다르게 그려질 수 있고, 또는 예를 들어, 콘테이너가 매우 작아서 아이콘들이 서로 중첩되거나 표현의 다른 특징들을 불명료하게 하여 생략하는 것이 편리하다면 생략될 수도 있다.

비고정된 에지들이 그려질 수 있는 정확한 위치는 콘테이너의 콘텐츠에 의존할 수 있다. 이후에 기술되는 바와 같이, '라이브 프루핑(live proofing)'이 사용되는데, 이는 콘텐츠가 문서 템플릿과 통합되어 사용자 인터페이스에서 보이는 것을 의미한다. 대안적인 구현에는, 모든 문서들에 대해 평균이 되는 콘테이너의 콘텐츠 영역 및 비고정된 에지들이 사용자 인터페이스에 배치되어야 하는 위치를 결정하는 다른 수단을 사용할 수 있다.

이러한 콘테이너들의 표현들은 콘테이너 에지 상태들을 디스플레이하는 그래픽 방법을 제공한다. 표현들의 해석은 다음과 같다:

● 점선은 문서들 내의 에지가 콘테이너의 콘텐츠에 의존하는 위치를 의미한다. 도 4에서, 이러한 에지는 410으로 라벨링된다.

● 실선은 에지가 (에지들(414)과 같이) 고정되거나 콘테이너의 폭 또는 높이가 고정되기 때문에(둘 모두 콘테이너(408)에 고정됨), 에지들이 제한됨을 의미한다.

● 앵커는, 앵커가 고정된, 교차하는 에지들 및/또는 축들을 의미한다. 그러므로, 앵커 점은 모든 문서들에서 동일한 수평 및 수직 위치에 나타낼 것이다. 그러므로, 앵커가 정의에 의해 고정된다. 도 4에서의 아이콘(409)은 교차하는 에지들(414)이 고정됨을 나타내는 앵커 아이콘의 예이다.

● 슬라이더는 연관된 에지가 고정됨을 의미하지만 콘테이너는 에지를 따라 '세로로 슬라이딩하는' 많은 위치들에 위치될 수 있다. 예를 들어, 도 4에서, 슬라이더들(413)은 문서에서 특정 도면들에서 보여지는 위치의 좌측 또는 우측에 나타날 수 있다.

어떤 툴 또는 어떠한 콘테이너들이 선택되거나, 강조되거나 또는 그렇지 않으면 활성화되었는지에 의존하여, 이들 아이콘들 또는 에지들의 일부 또는 모두가 그려지거나 그려지지 않을 수 있다. 콘테이너 에지들 및 아이콘들은 일반적으로 문서 템플릿의 설계에만 도움을 주기 때문에 프린트된 문서에 나타나지 않는다.

최소 및 최대 폭 및 높이와 같은 세팅들이 2차 다이얼로그 윈도우에서 디스플레이될 수 있다.

도 5a에서, 콘테이너(501)는 폭 및 높이 둘 모두에서 비고정된다. 고정된 에지들(503)은 실선들로 표시된다. 비고정된 에지들(504)은 파선들로 표시된다. 확대 아이콘들(505)은 인접 에지들(504)이 비고정된 부가적이거나 대안적인 표시자들이다.

도 5b에서, 콘테이너(501)는 폭 및 높이 둘 모두에서 비고정된다. 앵커 아이콘(506)은 해당 아이콘에서 교차하는 둘 모두의 에지들(503)이 고정됨을 부가적으로 또는 대안적으로 나타낸다.

도 5c에서, 콘테이너(501)는 선택적 앵커 아이콘(507)에 의해 표시된 바와 같이 중심점 주변에서 동등하게 발생하는 콘테이너의 확장 또는 수축으로 폭 및 높이 둘 모두에서 비고정된다.

도 5d에서, 콘테이너(501)는 상부 에지(508)가 고정되는 것을 제외하고 폭 및 높이 둘 모두에 비고정된다. 앵커 아이콘(509)은 아이콘이 위치하는 중심에 상부 에지(508)가 고정됨을 나타내며, 또한 그 콘테이너의 좌측 및 우측 에지들이 그 아이콘을 통해 수직으로 그려진 중심축선(수직축) 주위로 확장 또는 수축된다.

6.3.3 콘테이너 속성들의 그 자리 편집을 위한 방법

마우스(133) 및 포인팅 디바이스(313)로 임의의 에지(503 또는 504) 부근을 클릭함으로써, 에지(503/504)는 고정된 상태와 비고정된 상태 간에 전환되고, 그에 따라 사용자 인터페이스(301) 내의 그래픽 표현이 갱신된다.

제어 점들(502)은 마우스(133)와 포인팅 디바이스(133)의 조합을 사용하여 드래그될 수 있는데, 이로 인해 대응하는 에지 또는 에지들이 제어 점(502)의 위치에따라 변경된다. 마우스(133)/포인터(313)에 의한 선택으로부터 제어 점(502)을 해제하면, 새로운 위치가 무효가 되도록하는 어떠한 외부 제한들도 존재하지 않는다고 가정했을 때, 고정된 에지는 드래그된 위치에 남는 반면, 비고정된 에지는 계산된 위치로 돌아가는데, 계산된 위치는 콘테이너 내부의 콘텐츠의 크기 및 모양과 같은 인자들 및 콘테이너 또는 에지에 적용할 수 있는 임의의 다른 제한들에 의해 결정된다. 비고정된 에지들 및 콘테이너들의 위치를 결정하는데 사용된 알고리즘들의 완전한 설명에 대해서는 이하를 참조한다.

두 개의 비고정된 에지들의 교차점 또는 비고정된 에지의 중심에 위치된 제어 점들은 고정되거나 전혀 보이지 않을 수 있으며, 비고정된 에지의 어느 한쪽에 위치된 제어 점들은 에지에 평행한 방향으로의 이동으로 제한될 수 있다.

부가적으로, 꼭지점 또는 에지에 근접하여, 또는 꼭지점에 위치하는 제어 점 상에, 또는 에지의 중심에 포인팅 디바이스(133)를 클릭함으로써, 앵커들을 콘테이너의 꼭지점 또는 에지에 부가할 수 있다. 또한, 앵커 아이콘에 근접하여 클릭하거나, 대안으로 앵커를 선택하고 이어서 메뉴 아이템, 버튼 또는 인터페이스 내의 다른 제어를 활성화함으로써, 앵커들을 제거할 수 있다.

이러한 방법으로 앵커를 부가함으로써 앵커를 교차하는 에지, 에지들, 축 또는 축들이 고정될 것이다.

이러한 요구조건 없이도 수평 또는 수직 위치가 열악하게 정의되어 있는 콘테이너들을 생성할 수 있기 때문에, 콘테이너들은 고정된 적어도 하나의 점을 갖는 것이 바람직하며, 그렇지 않으면 외부적으로 제한된다. 예를 들어, 어떠한 고정된 에지들 또는 축들도 갖지 않는 콘테이너의 콘텐츠는, 그 콘테이너가 한편으로는 동작 가능한 어떠한 외부 제한들도 갖지 않는다면 논리적으로는 페이지 상의 어디에서나 나타날 수 있다. 이는 사용자가 그러한 콘테이너의 콘텐츠가 문서들 내에 나타날 곳을 예측하기가 어렵게 한다. 결과적으로, 일부 구현예들은 이처럼 열악하게 정의된 상태에서 콘테이너들을 배치할 수 있는 변경을 허용하지 않음으로써 이러한 가능성을 방지한다

상술한 바와 같이, 콘테이너가 외부 제한을 갖는다면, 콘테이너는 대응하는 고정된 에지들을 갖지 않고도 고정된 폭 또는 높이를 가질 수 있다. 예를 들어, 콘테이너의 수평 위치를 몇몇 수평 제한에 의해 확인할 수 있으면, 콘테이너는 비고정된 좌측 및 우측 에지들을 가짐에도 불구하고 고정된 폭을 가질 수 있다. 그러한 외부 제한들은 아래에 논의되며, "스트럿들(struts)"이라고 불린다.

외부 제한 스트럿이 콘테이너에 부착되면, 에지들을 고정하기 위해 앞서 기재된 것과 유사한 방식으로 에지들을 클릭함으로써 폭 또는 높이를 고정할 수 있다. 이러한 환경에서, 비고정된 좌측 또는 우측 에지 상에, 또는 그 근처에 클릭함으로써 콘테이너에 접속된 수평 스트럿이 있으면 그 콘테이너의 폭을 고정시킬 것이며, 비고정된 상부 혹은 하부 에지 상에, 또는 그 근처에 클릭함으로써 그 콘테이너에 접속된 수직 스트럿이 있으면 그 높이를 고정시킬 것이다.

에지, 축, 폭 및 높이 고정을 조작하는 것은 콘테이너에 대한 에지 상태들 간의 변화에 대응한다. 고정된 에지 또는 코너의 위치를 변경하는 것과 같은 다른 조작들은 다른 콘테이너 상태 정보를 변경한다.

바람직하게, 각각의 콘테이너가 항상 유효한 에지 상태를 가짐을 보장하기 위해, 제한이 콘테이너로부터 제거되거나 추가될 때마다 한 세트의 규칙들이 적용되고, 임의의 문제들을 해결하기 위해 다른 제한들에 필요한 조정들이 수행된다. 조정들은 새로운 제한 구성을 보여주기 위해 스크린 디스플레이를 업데이트함으로써 사용자에게 즉시 표시된다. 이로 인해 사용자 인터페이스(103)는 콘테이너들이 바람직하지 않은 상태를 갖는 것을 방지하게 된다.

규칙들은 다음과 같이 각각의 콘테이너의 수평 및 수직 차원들에 개별적으로 적용된다:

1. 임의의 에지가 외부 제한에 의해 고정되거나 동작되면, 그 에지에 평행한 축은 비고정된다.

2. 축이 비고정되면, 적어도 하나의 평행한 에지가 외부 제한에 의해 고정되거나 동작되어야 한다.

3. 대향하는 에지들이 비고정되고 그 에지들 상의 모든 외부 제한들이 제거되면, 그 에지들에 평행한 축은 고정된다.

4. 축이 고정되면, 그 축에 평행한 에지들은 비고정되며 그 에지들 상에서 동작하는 외부 제한들은 제거된다.

5. 폭이 고정되면, 좌측 및 우측 에지들 및 수직축은 비고정된다.

6. 높이가 고정되면, 상부 및 하부 에지들 및 수평 축이 비고정된다.

6.3.4 대안적인 콘테이너 사용자 인터페이스: 개별 제한들

콘테이너 사용자 인터페이스의 대안적인(제2의) 구현은, 콘테이너의 에지 상태를 시각적으로 식별하고, 개별적으로 편집 가능한 제한들로 구분하는 방식으로 콘테이너들의 편집을 디스플레이 및 허용할 수 있다.

이러한 구현예에서, 폭 및 높이 제한들은 콘테이너의 콘텐츠 영역을 가로지르는 바들로 표시된다. 에지들은 단지 에지의 고정된 또는 비고정된 성질을 나타낸다.

콘테이너 디스플레이 방법에 따른 제2의 구현에 대한 규칙들은, 차례로:

1. 각각의 에지 또는 축에 대해, 선이 고정되면 실선으로 그려지고, 그렇지 않으면 파선으로 그려지며;

2. 폭이 고정되면, 콘텐츠 영역을 가로지르는 바를 그리고, 그렇지 않으면 파선으로 그리며;

3. 높이가 고정되면, 콘텐츠 영역 아래로 실선인 높이 바를 그리고, 그렇지 않으면 파선으로 그리고;

4. 에지들 및/또는 축들에 수직인 각각의 쌍에 대해, 둘 모두 고정되면 그들의 교차점에 앵커를 그리고, 그렇지 않으면 핸들을 그리는 것이다.

상술한 바와 같이, 실선이 이미 슬라이더와 동일한 동작을 나타내기 때문에, 제2의 구현에서는 어떤 슬라이더도 필요없다. 해당 구현이 에지의 고정에 부가하여 폭 및 높이를 나타내기 위해 실선들을 사용하기 때문에, 이전의 구현에 대해서는 적용되지 않는다.

제2의 구현의 폭 및 높이 바들은 콘테이너 상의 폭 및 높이 제한들을 각각 디스플레이하고 편집하는 그래픽 수단이다. 표시된 고정 및 비-고정 각각에 대해 실선 또는 파선의 외관을 갖는 바들은 에지들처럼 보일 수 있다. 도 38에, 폭 바(3805) 및 높이 바(3806)가 두 개의 콘테이너들(3802, 3803)에 걸쳐있는 것으로 도시되는데, 그 수평 및 수직 위치들 각각은 콘테이너(3801)의 크기 및 접속 스트럿들(3804)의 길이에 의해 결정된다. 선택적으로, 고정된 에지들과 같은 다른 제한들이 폭 또는 높이를 고정하는 것을 무의미하게 한다면, 바들은 비활성화되거나 그려지지 않을 수 있다. 예를 들어, 콘테이너의 좌측 에지가 고정되고 우측 에지가 비고정되면, 콘테이너는 이전에 논의된 36가지의 바람직한 에지 상태들의 세트 외의 에지 상태에 존재하게 되므로, 사용자가 폭을 고정시키는 것을 방지하는 것이 바람직할 수 있다. 이러한 상황에서, 폭 바는 나타나지 않을 수 있다.

이러한 제2의 구현에의 콘테이너들을 편집하기 위해서, 마우스(133) 및 포인터(313)를 사용하여 에지, 축, 또는 폭 혹은 높이 바를 클릭하는 것은 에지를 고정된 것에서 비고정된 것으로, 또는 그 역으로 변화시킨다. 에지, 핸들 또는 앵커를 드래그하는 것은 기술한 바와 유사한 방식으로 동작하고, 앵커를 클릭하는 것은 교차하는 임의의 비고정된 에지들 또는 축들을 고정하며, 즉 둘 모두의 교차 선들이 이미 고정되어 있다면, 둘 모두 비고정이 된다. 이로 인해 콘테이너의 수평 및 수직 위치가 부족하게 제한된 채로 존재하게 된다면, 중심 축들 중 어느 하나 또는 둘 모두를 고정시킴으로써 이러한 문제를 해결할 수 있다. 예를 들어, 앵커를 클릭함으로써 콘테이너의 모든 에지들이 비고정으로 유지된다면, 다수의 가능한 콘텐츠 위치들이 존재할 수 있고, 따라서 이러한 문제를 해결하기 위하여 수직 또는 수평 축들 모두를 고정시킴으로써 콘테이너의 중점이 앵커될 수 있다. 다른 외부 제한들 또는 규칙들이 이러한 단계를 불필요하게 할 수 있고 또는 단지 하나의 축만을 고정하는 것만으로도 충분할 수 있다.

6.3.5 대안적인 콘테이너 사용자 인터페이스: 크기 프레임

또 다른 (제3의) 구현에서, 콘테이너의 동작은 최소 및 최대 범위들의 직접 조작에 의해 제어되며, 각각은 분리된 직사각형 프레임으로 표시된다.

도 6a 내지 도 6c는 콘테이너에 대한 제3의 예시적인 규칙들을 설명한다.

도 6a를 참조하면, 콘테이너(601)가 각각의 꼭지점 및 선택적으로는 각각의 에지의 중심에 배치된 제어 점들(603)을 갖는 최소 범위의 프레임(602)과, 각각의 꼭지점 및 선택적으로는 각각의 에지의 중심에 배치된 제어 점들(605)을 갖는 최대 범위의 프레임(604)을 포함한다.

최소 범위의 프레임(602)은 항상 수평 및 수직 차원들 둘 모두에서 최대 범위의 프레임(604) 내부 또는 그와 일치하여 나타난다.

제 3의 프레임(606)이 콘테이너 내부의 콘텐츠의 실제 범위들을 나타낸다. 콘텐츠 프레임(606)은 항상 최대 범위의 프레임 내부 또는 그와 일치하여 나타나고, 항상 최소 범위 프레임의 외부 또는 그와 일치하여 나타난다. 콘텐츠 프레임의 바람직한 크기는 콘테이너 내부의 콘텐츠의 크기 및 모양과 같은 인자들 및 콘테이너에 적용할 수 있는 임의의 다른 제한들에 의해 결정된다.

사용자가 세 개의 프레임들을 서로 쉽게 구별하기 위해, 각각의 프레임 및 제어 점들은 다른 선 두께, 스타일 또는 색상에 의해 선택적으로 표현될 수 있다.

하나 이상의 콘테이너를 포함하는 문서 템플릿을 보는 경우 시각적인 혼란을 감소시키기 위해, 최대 및 최소 범위 프레임들 및 그들의 연관된 제어 점들(602, 603, 604 및 605) 및 콘텐츠 프레임(606)은, 예를 들어, 콘테이너에 의해 점유된 영역 내에 포인터를 배치함으로써 또는 마우스로 콘테이너 위에 클릭함으로써 그들이 관련된 특정 콘테이너가 '활성화'되지 않으면, 선택적으로 숨겨질 수 있다.

포인팅 디바이스(133)을 사용하여 그들 각각의 제어 점들(603, 605)의 직접적인 조작에 의해 최소 범위 프레임(602) 및 최대 범위 프레임(604) 모두가 리사이징될 수 있다. 최소 범위 프레임(602)을 리사이징함으로써, 어떠한 콘텐츠가 콘테이너 내부에 나타날 수 있는지와 무관하게 사용자는 콘텐츠 프레임(606)의 가능한 최소 크기를 결정할 수 있다. 최대 범위 프레임(604)을 리사이징함으로써, 어떠한 콘텐츠가 콘테이너 내부에 나타날 수 있는지와 무관하게 사용자는 콘텐츠 프레임(606)의 가능한 최대 크기를 결정할 수 있다.

최소 범위 프레임(602)를 리사이징하는 경우, 프레임의 임의의 꼭지점 또는 에지의 위치가 최대 범위 프레임(604)의 외부로 드래그되면, 꼭지점 또는 에지 중 하나는 제어 점이 해제된 후 대응하는 최대 범위 프레임의 꼭지점 또는 에지와 동일하도록 자동으로 이동될 것이며, 꼭지점 또는 에지는 대응하는 최대 범위 프레임(604)의 꼭지점 또는 에지에 도달한 후에는 임의의 추가적인 이동을 중지하거나, 대응하는 최대 범위 프레임(604)의 꼭지점 또는 에지는 드래그되는 꼭지점 또는 에지와 일치하거나 또는 외부에 존재하도록 자동으로 확장될 것이다.

최대 범위 프레임(604)를 리사이징하는 경우, 프레임의 임의의 꼭지점 또는 에지의 위치가 최소 범위 프레임(602)의 내부로 드래그되면, 꼭지점 또는 에지 중 하나는 제어 점이 해제된 후 대응하는 최소 범위 프레임의 꼭지점 또는 에지와 동일하도록 자동으로 이동될 것이며, 그 꼭지점 또는 에지는 대응하는 최소 범위 프레임(604)의 꼭지점 또는 에지에 도달한 후에는 임의의 추가의 이동을 중지하거나, 대응하는 최소 범위 프레임(602)의 꼭지점 또는 에지는 드래그되는 꼭지점 또는 에지와 일치하거나 또는 그 내부에 존재하도록 자동으로 축소될 것이다.

최대 범위 프레임(604) 내의 콘텐츠 프레임(606)의 위치는 최대 범위 프레임(604) 내의 최소 범위 프레임(602)의 관련 위치에 따라서 결정된다.

최소 범위 프레임(602)은 포인팅 디바이스(133)로 프레임을 드래그함으로써 최대 범위 프레임(604) 내의 임의의 위치로 이동될 수 있다.

선택적으로, 최소 범위 프레임(602)은 포인팅 디바이스(133)로 드래그되는 동안 수평 차원으로 좌측, 우측 및 중심 그리고 수직 차원으로 상부, 하부 및 중심을 포함하는 최대 범위 프레임(604) 내의 바람직한 위치들의 선택에 가장 가까이에 '스냅(snap)'하도록 프로그램될 수 있다.

도 6a의 수평 차원들(a, b, c 및 d)은 최대 범위 프레임(604), 콘텐츠 프레임(606) 및 최소 범위 프레임(602)의 각각의 수직 에지들 간의 거리들을 나타낸다.

임의의 특정 시간에 프레임 내부의 콘텐츠에 의해 결정되는 바와 같이 그 콘텐츠 프레임(606)의 임의의 크기가 주어지면, 차원들(a, b, c 및 d)은 항상 다음 방정식에 따를 것이다:

a:b = d:c

대응하는 방정식이 대응하는 수직 차원들에 적용되고, 이는 도면들에 라벨링되지 않았다.

그러므로, 최대 범위 프레임(604) 내의 콘텐츠 프레임(606)의 위치는 나뉘어진 수평 크기들(a, b, c 및 d) 및 대응하는 수직 차원들에 따라 결정된다.

도 6b는 일 특정 경우를 도시하고, 최소 범위 프레임(602)은 최대 범위 프레임(604)의 상부 및 좌측에 위치되고, 그러므로 콘텐츠 프레임(606)은 또한 최대 범위의 상부 및 좌측에 위치된다.

도 6c는 또 다른 특정 경우를 도시하고, 최소 범위 프레임(602)는 수평 및 수직 차원 둘 모두에서 최대 범위 프레임(604) 내에서 중앙에 위치되며, 그러므로 콘텐츠 프레임(606)은 또한 수평 및 수직 차원 둘 모두에서 최대 범위 내의 중앙에 위치된다.

도 6d 및 도 6e는 콘테이너 범위들이 사용되는 가변 데이터 문서를 생성하는 방법(620)을 도시한다. 상기 방법(620)은 애플리케이션(121) 내에서 구현되고, 단계(624)에서 문서 템플릿이 디스플레이(144) 상의 GUI(301) 내에서 검색되고 디스플레이된 후에 시작 점(622)을 갖는다. 단계(626)는 통상적으로 마우스(133) 및 대응하는 포인터(313)의 움직임을 통해 사용자에 의해 콘테이너의 도시(drawing)를 검출한다.

그려진 콘테이너는 디폴트로서 도 6a 내지 도 6c에 도시된 바와 같이 최소 콘테이너 범위 및 최대 콘테이너 범위인 두가지 제한들을 갖는다. 콘테이너의 최초 상태에서, 최소 콘테이너는 최대 콘테이너 범위들과 일치한다. 단계(634)에서, 어느 하나의 범위의 제어 점들을 조작하여 콘테이너 범위를 변경할 수 있다.

여러가지 제한들이 단계(628)에 적용될 수 있는 레이아웃을 형성하기 위해 콘테이너가 템플릿 내에 표시된다. 단계(628)는 레이아웃 내의 콘테이너들 간에서 본 명세서에 도시된 바와 같이 많은 형태의 제한들의 통합에 포함될 수 있다. 현재 기재된 구현에서, 적용된 제한은 도 6a 내지 도 6c에 도시된 바와 같은 콘테이너 범위의 것이다. 단계(630)에서, 마우스(133) 및 포인터(313)를 사용하여 콘테이너 범위가 다시 그려지고, 단계(632)에서, 레이아웃 내의 콘테이너와 연관된다. 단계(634)에서, 범위의 제어 점들은 선택될 수 있고 콘테이너 범위를 변경하기 위해 조작될 수 있다. 그러한 변경은, 범위의 하나 이상의 에지들을 콘테이너의 에지들과 일치되도록 세팅하는 단계(도 6b)와, 연관된 콘테이너에 관련한 범위 프레임을 이동시키고 위치시키는 단계를 포함한다.

단계(628)로 돌아감으로써, 사용자가 추가적인 제한들을 적용할 수 있는 단계(636)가 계속된다. 추가적인 제한은 추가적인 범위 프레임을 포함함으로써, 최소 및 최대 범위 프레임 모두를 구성할 수 있는 기회를 제공한다. 또한, 다른 제한들도 적용될 수 있다. 예를 들어, 스트럿은 최대 범위 프레임과 연관되지 않는 추가적인 콘테이너와 최대 범위 프레임 사이에 적용될 수 있다. 이러한 경우에, 최대 범위 프레임과 연관된 콘테이너의 크기를 계속해서 제한하면서, 추가적인 콘테이너의 변화들에 따라 범위 프레임이 동적으로 이동하도록 허용할 수 있다.

단계(636) 후에, 단계(638)의 동작을 통해, 사용자는 레이아웃이 완료되는 시점까지 레이아웃에 추가적인 콘테이너들을 추가할 수 있다. 레이아웃이 일단 완료되면, 단계(640)는 문서를 생성하기 위해 레이아웃에 콘텐츠의 레코드를 배치한다. 단계(642)는 이러한 프로세스가 콘텐츠의 모든 레코드에 대해 반복될 수 있도록 하여 가변 데이터 문서의 세트를 생성한다. 일단 모든 레코드들이 소비되고 문서들이 생성되면, 단계(644)에서 요구되는 바와 같이, 이러한 점이 검증(섹션 11.10 내지 11.13 참조) 및/또는 프린트될 수 있다. 방법(620)은 단계(646)에서 종료된다.

7. 이미지 콘테이너들

이미지 콘테이너는 사진, 그림, 로고 또는 도면과 같은 이미지를 포함하기 위한 콘태이너의 특정 형태이다.

바람직하게는, 이미지 콘테이너들은 포함될 이미지의 스케일링을 제어하기 위해 다음의 동작들의 선택 중 하나를 가질 수 있다:

● 전체 이미지가 콘테이너 내에 맞추어지고 이미지의 영상비가 유지되도록 스케일링 업 또는 다운되는 '전체 이미지 맞추기(Fit entire image)'

● 전체 콘테이너가 이미지로 채워지고 이미지의 영상비가 유지되어 이미지의 임의의 잔여 부분은 뷰로부터 잘려지는 '이미지를 박스의 크기에 맞추기(Fit image to size of box)'

● 수평 및 수직 차원 모두로 콘테이너의 크기를 정확히 맞추기 위해 스케일 업 또는 다운되며, 그 이미지의 원래 영상비를 무시하는 '맞추어 확장(Stretch to fit)'

● 어떠한 스케일링도 이미지에 적용되지 않고 콘테이너의 외부에 있는 이미지의 임의의 부분들이 뷰로부터 잘려지는 '스케일링 하지 않기(Do not scale)'.

부가적으로, 수평 차원에서 좌측, 중심 또는 우측 그리고 수직 차원에서 상부, 중심 또는 하부의 임의의 조합히 되도록, 콘테이너 내부에 있고 콘테이너보다 작은 임의의 이미지가 특정 방향으로 정렬되도록 이미지 콘테이너들을 세팅할 수 있다.

8. 텍스트 콘테이너들

텍스트 콘테이너는 여러가지 폰트 및 단락 스타일로 포맷될 수 있고 텍스트 콘테이너의 여러가지 에지들에 정렬되거나 맞춰질 수 있는 텍스트의 범위를 포함하기위한 콘테이너의 특정 타입이다.

텍스트 콘테이너들은 정적 텍스트, 가변 텍스트, 또는 그 둘의 조합 중 하나를 포함한다. 이미지들과 같은 다른 대상들도 또한 텍스트 콘테이너들에 삽입될 수 있고, 동일한 방식의 텍스트 흐름에 따라 흐를 것이다.

정적 텍스트는 콘테이너 내에 직접적으로 타이핑함으로써 입력된다. 가변 텍스트는 섹션 11에 보다 상세히 설명되는 바와 같이 콘테이너 내로 라이브러리로부터의 가변 데이터 대상을 드래그함으로써 부가된다. 하나 이상의 가변 텍스트 대상이 단일의 텍스트 콘테이너에 디스플레이될 수 있다.

정적 텍스트의 포맷팅은 개별 문자들, 워드들 또는 전체 단락들에 적용될 수 있지만, 옵션들을 포맷팅하는 것은 단지 가변 텍스트 아이템의 전체 경우에만 적용될 수 있다.

8.1 텍스트 포맷팅

텍스트 포맷팅은 다음 옵션들을 포함한다:

- 폰트
- 폰트 크기
- 두껍게
- 이탤릭체
- 밑줄
- 색상
- 줄 간격
- 장평(Force Capitalization)
- 자동 하이픈 넣기

8.2 정렬

다음 텍스트 정렬 옵션들이 몇가지 특정 구현들에서 허용된다:

- 수평 정렬: 좌측(디폴트), 수평으로 가운데, 우측 또는 우측정렬.
- 수직 정렬: 상부(디폴트), 수직으로 가운데, 하부.

다른 옵션들이 수직으로 정렬된 텍스트와 같은 다른 구현들에 적절할 수 있다.

8.3 열들

텍스트 콘테이너 내의 텍스트는 각각의 열의 좌측 및 우측 에지들을 각각의 인접한 열로부터 분리하는 '거터(gutter)'로서 알려진 공간을 이용하여, 단일의 열 또는 둘 이상의 수직 열들로 배치될 수 있다.

인접한 열들 간의 분할 선을 드래그함으로써 열의 폭을 조정하는 것이 종래 기술에서 보편적인 반면, 거터들은 속성 시트 또는 대화 박스에서 값을 변경함으로써 또는 유사한 비-직접적인 방법에 의해 이미 세팅되었다.

도 7a 내지 도 7b는 세 개의 열을 갖는 텍스트 콘테이너(701) 및, 열 및 거터 폭들 모두가 마우스(133) 및 포인터(313)와 같은 포인팅 디바이스를 사용하여 직접적인 조작에 의해 리사이징될 수 있는 방법을 도시한다.

도 7a를 참조하면, 텍스트 콘테이너(701)는 텍스트의 세 개의 열들(702)로 분리된다. 도시된 바와 같이, 선들(703)은 인접한 열들 간의 중심 분리 선들을 나타내고, 선들(704)은 인접한 열들(702) 간의 거터 경계들을 나타낸다.

마우스(133) 및 포인팅 디바이스(133)로 중심 분리선(703)을 드래그함으로써, 인접한 열들의 폭이 조정될 수 있다. 열 드래그 기능이 활성임을 나타내기 위해, 드래그 기능 동안 디폴트 포인터 대신에 특정 포인터(705)가 디스플레이된다.

도 7b에 도시된 바와 같이, 포인팅 디바이스로 거터 경계선(704)를 드래그함으로써, 거터의 폭이 조정될 수 있다. 거터 드래그 기능이 활성임을 나타내기 위해, 드래그 기능 동안 디폴트 포인터 대신에 특정 포인터(706)가 디스플레이된다.

동일한 열 간격 내의 모든 거터 경계선들은, 거터 경계선들과 중앙 분리선 각각의 사이에 동일한 간격이 항상 유지될 수 있도록 동시에 조정된다.

또한 도 7b에 도시된 바와 같이, 둘 이상의 열들이 있는 경우, 각각의 거터는 개별적으로 조작될 수 있다.

8.4 가변 폰트 크기

텍스트의 다양한 부피(volume)들을 줄이거나 늘림으로써 콘테이너 내부에 맞추어지도록 하고, 콘테이너의 크기가 외부 제한들에 따라 변경되도록 하는 한편, 여전히 텍스트가 전체적으로 콘테이너 내부에 맞추어지도록, 텍스트 콘테이너 내부의 텍스트의 폰트 크기가 변경될 수 있다. 최소 및 최대 폰트 크기가 폰트 크기들의 가능한 범위를 제한하는 각각의 텍스트 콘테이너에 대해 설정될 수 있다. 콘테이너 내부에 모든 텍스트를 맞추기 위해서 사용할 최적의 폰트 크기를 결정하기 위하여 바이너리 검색 알고리즘이 사용될 수 있다. 폰트 크기가 최소 크기에 도달한 경우, 텍스트는 여전히 콘테이너에 맞지 않을 것이고, 사용자에게 에러가 나타날 것이다.

8.5 콘테이너들 간의 폰트 크기 동기화

임의의 텍스트 콘텐츠가 내부에 적절하게 맞추어지도록 하기 위해 콘테이너들 중 임의의 콘테이너의 폰트 크기가 수정되는지에 관계없이 폰트 크기가 모든 콘테이너들에서 동일해야 함을 지정하는, 둘 이상의 텍스트 콘테이너들 간의 제한이 생성될 수 있다.

바람직하게는, 제한의 추가에 앞서 이러한 동기화를 위해 선택된 텍스트 콘테아너들이 다른 폰트 크기들을 갖는다면, 폰트 크기들은 먼저 평균화되어 각각의 선택된 콘테이너에 적용된다. 다른 대안적인 구현에는 최초로 선택된 콘테이너의 폰트 크기를 사용하여 모든 다른 다른 콘테이너들에 동일한 폰트 크기를 적용할 수 있다. 또 다른 대안은 각각의 콘테이너에 외부 폰트 크기 세팅을 적용하는 것이다.

도 8을 참조하면, 애플리케이션 윈도우(301)는 앞서 기재된 바와 같이, 툴바 영역(303)을 갖는다. 툴바 영역(303)은 폰트 크기 동기화 제한을 부가하기 위해 적어도 폰트 선택기(803), 폰트 크기 선택기(804) 및 버튼(805)을 포함한다. 폰트 선택기(803) 및 폰트 크기 선택기(804)는 텍스트 콘테이너 내의 텍스트의 속성들을 바꾸는데 사용될 수 있다. 몇몇 텍스트 콘테이너들이 선택된 채로, 폰트 크기 동기화 버튼(805)을 클릭함으로써 모든 선택된 콘테이너들의 폰트 크기들이 동기화될 것이다.

복수의 텍스트 콘테이너들(806, 807)이 문서 템플릿(309)에 사전에 배치되어 있다. 포인터(313)를 이용하여 콘테이너들을 클릭하거나, 텍스트 콘테이너들 주위로 선택 직사각형을 드래그하거나, 또 다른 방법을 통해 마우스(133)를 조작함으로써 콘테이너들을 선택할 수 있다. 텍스트 콘테이너가 현재 선택되어 있음을 나타내기 위하여, 콘테이너들 상에 제어점들을 보여주는 것과 같은 시각적 수단을 사용할 수 있다.

둘 이상의 텍스트 콘테이너들(806 및 807)을 선택하고, 동기화 버튼(805)을 활성화함으로써 제한이 추가된다. 다른 구현에서, 폰트 크기 동기화 제한은 풀 다운 메뉴, 키보드 명령 또는 다른 수단들에 의해 적용될 수 있다.

제한은 선택적으로 각각의 텍스트 콘테이너들(806 및 807) 상의 또는 근처의 아이콘(808) 또는 다른 그래픽 표현에 의해 사용자 인터페이스에 표시될 수 있다. 부가적으로, 텍스트 콘테이너들이 서로에 링크되었음을 나타내기 위해, 제한들에 의해 링크된 텍스트 콘테이너들의 각각의 아이콘들 사이에 결합선(joining line)(809)을 그릴 수 있다.

다른 구현들에서, 선택된 대상들의 여러가지 속성들을 디스플레이하기 위해 비디오 디스플레이 스크린(144)의 분리 영역을 제공할 수 있으며, 이러한 영역은 선택된 텍스트 콘테이너가 그에 적용된 폰트 크기 동기화 제한을 가짐을 나타내고, 다른 콘테이너들이 제한들에 링크됨을 나타내는데 사용될 수 있다.

폰트 크기 제한은, 제한에 의해 링크된 하나 이상의 텍스트 콘테이너를 선택하고 초기에 제한을 적용하는데 사용된 명령 또는 시퀀스를 반복함으로써 삭제될 수 있다. 이는 동기화 버튼(805)을 활성화하거나, 제한에 의해 링크된 하나 이상의 텍스트 콘테이너들을 선택하고 그 목적을 위해 특별히 제공된 부가적인 명령 또는 시퀀스를 적용하거나, 또는 대표 아이콘(808) 또는 결합선(809)을 클릭하고 키보드(132) 상의 또는 디스플레이(144) 상에 도시된 메뉴 명령으로부터의 'delete' 키의 적용과 같은 명령 또는 시퀀스를 적용함으로써, 달성될 수 있다.

사용자가 이미 다른 폰트 크기를 갖는 텍스트를 포함하는 둘 이상의 텍스트 콘테이너들에 폰트 크기 제한을 적용하면, 가장 큰, 가장 작은 또는 평균 폰트 크기 중 하나가 모든 텍스트 콘테이너들에 적용될 것이다. 대안으로, 팝업 대화 박스 또는 그와 유사한 것으로 폰트 크기를 선택하도록 사용자를 유도할 수 있다.

사용자가 폰트 크기 제한에 의해, 예를 들어, 폰트 크기 선택기(804)로부터의 새로운 폰트 크기를 선택함으로써, 이미 링크된 임의의 콘테이너들에 대해 그 폰트 크기를 수동으로 변화시킨다면, 제한에 의해 링크된 모든 콘테이너들에 새로운 폰트 크기가 적용된다.

8.6 내부 마진들(internal margins)의 자동 적용(automatic application)

텍스트 콘테이너의 내부 마진은 텍스트 콘테이너가 시각적 경계 및/또는 시각적 배경을 갖는지의 여부에 기초하여 자동으로 세팅될 수 있다.

도 9를 참조하면, 애플리케이션 윈도우(301)는 상술한 바와 같이 툴바 영역(303)을 갖는다. 툴바 영역(303)은 적어도 경계 선택기(903) 및 배경색 선택기(904)를 포함한다.

텍스트 콘테이너(905)는 어떠한 시각적인 경계 또는 배경색을 갖지 않으며, 또한 콘테이너의 에지들과 콘테이너 내부의 텍스트 간의 어떠한 마진도 존재하지 않는다.

텍스트 콘테이너(906)는 시각적 경계를 가지며, 콘테이너의 에지들과 모든 네 개의 측면 상의 콘테이너 내부의 텍스트 간의 마진(907)을 갖는다.

도 9에 도시된 바와 같이 시스템의 상태에 선행하는 이벤트들에서, 어떠한 시각적 경계 또는 배경도 갖지 않고 어떠한 내부 마진도 갖지 않는 텍스트 콘테이너(905)와 유사한 상태의 텍스트 콘테이너(906)가, 포인터(133)로 콘테이너(906)을 클릭함으로써, 또는 그 주위로 선택 직사각형을 드래그함으로써, 또는 다른 방법에 의해서 선택된다. 콘테이너 상에 제어점들을 보여주는 것과 같은 시각적 수단에 의해 텍스트 콘테이너가 현재 선택되어 있음을 나타낼 수 있다.

이어서 시각적 경계가 경계 선택기(903)을 사용하여 적용되거나, 대안으로 배경색이 배경색 선택기(904)를 사용하여 적용된다.

시각적 경계 또는 배경이 텍스트 콘테이너에 적용되는 경우에, 어떠한 내부 마진도 없다면, 미리 결정된 내부 마진(907)이 모든 네 개의 측면들 상의 텍스트 콘테이너에 자동으로 적용된다. 텍스트 콘테이너의 배경색과 템플릿의 우세한 주변 배경색 간의 시각적 식별이 있을때마다, 내부 마진이 자동으로 추가된다.

내부 마진이 콘테이너에 대해 자동으로 세팅될 수 있는 또 다른 상황은 콘테이너의 배경색이 페이지 색상과 다른 경우이다. 이 경우, 콘텐츠가 콘테이너의 에지까지 흘러 넘치도록 허용하기 보다는, 콘텐츠와 페이지 배경 간의 시각적 분리를 제공하도록 내부 마진이 추가될 수 있다. 콘테이너의 배경색이 완전한 투명이면, 정의에 의해 콘텐츠들은 페이지와 동일한 배경색으로 그려질 것이고 그래서 어떠한 내부 마진도 추가될 필요가 없다.

8.7 콘테이너들 간의 텍스트 흐름

텍스트의 크기가 이전의 콘테이너의 크기를 초과하면, 둘 이상의 텍스트 콘테이너들은 텍스트가 하나의 콘테이너로부터 다음의 콘테이너까지 흐르도록 하기 위해서 순차적으로 함께 링크될 수 있다.

텍스트가 모든 링크된 콘테이너들의 결합된 영역에 맞지 않는 경우, 다른 콘테이너들의 폰트 크기들 간의 관계를 유지하려고 노력하면서, 텍스트가 맞추어질 수 있는 각각의 콘테이너에 대한 폰트 크기를 찾을 필요가 있다.

링크된 콘테이너들의 세트에 대해 이를 달성하기 위한 한가지 바람직한 방법은 텍스트가 정확히 일치할 때까지, 또는 모든 링크된 콘테이너들이 그들 각각의 최소 폰트 크기보다 작은 폰트 크기에 도달할 때(에러 또는 사용자가 그 동작을 완료할 수 없다는 다른 표시가 존재하는 지점)까지, 각각의 콘테이너에 대한 폰트 크기들을 스케일링하기 위해 바이너리 검색 알고리즘을 사용하는 것이다. 하나의 콘테이너가 다른 것들 보다 앞서 최소의 폰트 크기에 도달하면, 이로 인해 알고리즈이 필수적으로 종료되는 것이 아님에 주의해야 한다. 이러한 경우에, 알고리즘은 콘테이너들 간의 폰트 크기 관계들을 유지하려고 노력하지만, 이를 보증하는 것은 아니다.

이를 달성하기 위한 대안적인 방법은 상기와 유사한 방법을 사용하는 것이지만, 이러한 방법은 임의의 링크된 콘테이너들이 최소 폰트 크기보다 작은 폰트 크기에 도달하면 에러 또는 다른 통지와 함께 종료된다. 이러한 대안적인 방법은 링크된 콘테이너들 간에 관련 폰트 크기들을 유지하지만, 이진 검색 알고리즘보다 더많은 경우에 해결책을 제시하는데 있어 에러 및 실패를 할 수 있다. 모든 링크된 콘테이너들 간의 폰트 크기 동기화가 요구되는 텍스트 흐름을 구현하기 위하여 대안적인 방법이 사용될 것이다.

폰트 축소를 이용하여 텍스트 흐름을 달성하는 또 다른 방법은, 텍스트가 콘테이너들의 세트에 맞을 때까지 또는 제1 콘테이너가 최소 폰트 크기에 도달할 때까지, 링크된 세트 내의 제1 콘테이너의 폰트 크기를 단순히 감소시키는 것이다. 제1 콘테이너가 최소 폰트 크기에 도달하면, 텍스트가 맞을 때까지 또는 모든 콘테이너들이 최소 폰트 크기에 도달할 때까지 제2 콘테이너를 줄이는 작업을 계속한다. 텍스트가 여전히 맞지 않으면, 어떻게 해서든 에러 통지와 함께 이를 표시한다. 이러한 방법에 있어서, 콘테이너들의 순서가 중요하며, 이는 텍스트가 흐르는 순서로서 정의되고, 따라서 텍스트가 콘테이너 A에서 시작하고 콘테이너 B에서 계속되면, A는 제1 콘테이너이고 B는 제2 콘테이너이다.

9. 가이드들

가이드는 콘테이너들 및 다른 가이드들의 위치 선정을 돕기 위해 페이지의 폭 또는 높이를 스패닝하는(spanning) 수직 또는 수평 라인이다. 고정 및 비고정(또는 플로팅의) 두 종류의 가이드가 존재한다. 가이드들은 스트럿들(이하 기술될 거리 제한들)을 통해 다른 가이드들 또는 콘테이너들에 접속될 수 있다. 가이드들은 문서에 나타나지 않고 단지 설계를 돕는 문서 템플릿(309)에 나타난다. 가이드들은 작업 영역 내의 눈금자 영역으로부터 포인팅 디바이스를 드래그함으로써 생성될 수 있다.

9.1 고정된 가이드들

고정된 가이드는 프린트 가능 영역과 같은 페이지 또는 페이지의 몇몇 부분에 고정된다. 고정된 가이드는 모든 문서들에 대해 동일한 위치를 갖는다는 점에서 고정된다. 이러한 가이드는 페이지 또는 페이지의 프린트 가능한 부분의 특정 에지에 관해, 예를 들어, 좌측 에지에 관해 고정될 수 있다. 이러한 예에서, (예를 들어 다른 용지 크기로 프린트하기 위해) 페이지가 리사이징되었다면, 가이드는 좌측 에지로부터 동일한 거리를 유지할 것이지만, 임의의 다른 에지로부터 필수적으로 동일한 거리를 유지하는 것은 아니다.

9.2 비고정된 가이드들

비고정된 또는 플로팅 가이드는, 문서 내의 데이터가 변화할 때, 예를 들어, 문서 템플릿이 데이터 소스로부터의 레코드와 통합될 때, 다른 문서들에서 다른 위치를 가질 수 있다. 이러한 예에서, 데이터가 문서 템플릿과 통합되는 경우, 하나 이상의 콘테이너들은, 비고정된 가이드의 위치가 스트럿들의 동작을 통해 결정되도록 할 수 있는 데이터 소스로부터의 데이터를 유지할 수 있다. 그러한 데이터가 문서 템플릿과 통합될 때까지, 비고정된 가이드의 위치가 알려지지 않거나, 또는 편집

및 설계를 위한 초기 (가능한 임의의) 위치가 제공될 수 있다. 특정 구현예에서, 데이터 소스로부터의 데이터가 문서 템플릿과 통합되는 경우, 플로팅 가이드는 항상 사용자에 의해 편집되거나 레이아웃 엔진(105)에 의해 변경될 수 있는 위치를 갖는다.

9.3 가이드 구현들

특정 구현예에서, 고정된 가이드와 비고정된 가이드 둘 모두는 작업 영역, 페이지 경계 또는 페이지 프린트가능 영역의 에지에서 눈금자들까지 확장할 수 있는 실선으로 표시된다. 비고정된 가이드와 고정된 가이드를 식별하는데 색상이 사용된다. 대안적으로, 고정된 가이드가 실선으로 표시될 수 있는 반면, 비고정된 가이드는 파선 또는 점선으로 표시될 수 있다.

도 12를 참조하면, 눈금자(308)의 수평 부분으로부터 포인터(133)를 이용하여 드래그함으로써 비고정된 가이드(1204)가 GUI(301)에 이전에 생성되었다. 제1 콘테이너(1201)는 비고정 에지(1202)를 갖는다. 스트럿(1203)이 에지(1202)로부터 비고정된 가이드(1204)로 생성된다. 제2 콘테이너(1205)가, 가이드(1204)에 직접적으로 정렬된 콘테이너(1205)의 상부 에지에 의해, 가이드(1204)에 직접적으로 부착된다. 제3 콘테이너(1206)가 스트럿(1207)에 의해 가이드(1204)에 접속된다. 또한, 스트럿(1209)에 의해 제4 콘테이너(1208)가 가이드(1204)에 접속된다.

비고정된 에지(1202)가 콘테이너(1201)의 높이 변화의 결과로서 이동하기 때문에, 가이드(1204)는 스트럿 제한(1203)을 유지하기 위해 수직 위치를 변경할 것이다. 결과적으로 가이드(1204)와 관련하여 콘테이너들(1205, 1206 및 1208)은 그들의 제한들을 유지하기 위해 이동할 것이다. 유사하게, 문서 템플릿으로 데이터를 통합하는 결과로서, 콘테이너 크기들이 결정되거나 변경된다면, 스트럿들의 동작은 비고정된 가이드들이 콘테이너들 내의 콘텐츠들에 기초하여 위치들을 확보하도록 할 수 있다.

콘테이너와 가이드 간에 스트럿 제한을 부가하기 위한 방법은, 다른 설명 섹션들에 기재된 콘테이너들 간에 스트럿을 부가하기 위한 방법과 유사하다.

예컨대, 도 12에서의 콘테이너(1205)처럼, 콘테이너의 에지가 가이드에 직접적으로 부착되도록하는 제한을 부가하는 하나의 바람직한 방법은, 가이드에 시각적으로 '스냅'되는 지점인 가이드로부터의 소정 거리 내의 위치로 콘테이너 또는 콘테이너의 에지를 드래깅하거나, 이와 유사하게 콘테이너의 에지로 스냅되는 지점인 콘테이너 에지로부터의 소정 거리 내로 가이드를 드래깅하는 것이다. 콘테이너, 에지 또는 가이드가 스냅핑된 상태에서 드래그 동작으로부터 해제되면, 제한이 생성될 것이다.

10. 스트럿들

스트럿은 콘테이너들의 특정 콘테이너의 에지들과, 가이드들 및 가장자리들과 같은 개체들 사이에 고정된 거리를 유지하기 위하여 사용되는 제한의 일종이다.

바람직하게는, 각각의 스트럿은, 동일한 문서 템플릿 내의 두 개의 다른 콘테이너들에 속하는 정확히 두 개의 평행 에지들에, 또는 콘테이너의 한 에지와 평행 가이드에, 또는 두 개의 평행 가이드들 사이에 적용된다. 다른 구현예들은 스트럿들이 접속할 수 있는 것을 제한하거나, 스트럿들이 폭 또는 높이 제한들을 지정하는 방식으로서 단일의 콘테이너의 에지들을 접속하도록 허용하거나, 또는 스트럿들이 에지들 또는 가이드들을 평행한 페이지 마진들 또는 에지들과 접속시키도록 허용할 수 있다.

스트럿들은, 스트럿에 의해 부착된 에지들 중의 어떤 것도 배치의 면에 있어 다른 에지에 대해 우선순위를 갖지 않는다는 의미에서 대칭이다.

스트럿의 길이는 시각적으로 또는 수치적으로 또는 양자 모두의 조합에 의해 표현될 수 있다. 일정한 구현예에서, 모든 스트럿들은 그래픽 표현(스트럿 아이콘)의 길이에 의해 시각적으로 그들의 현재 길이를 디스플레이하는 반면, 다른 속성들은 대화 박스 내에서 숫자 또는 시각적 세팅들로서 시각화된다.

도 4를 참조하면, 제1 콘테이너(407)의 에지(410)가 스트럿 제한에 의해 제2 콘테이너(408)의 에지(411)에 접속된다. 스트럿 제한은 여러가지 애플리케이션의 모드들, 예를 들어 미리보기 모드에서 선택적으로 숨겨질 수 있는 스트럿 아이콘(412)으로 표시된다.

스트럿 아이콘들은 보통 일반적인 동작 모드에서 숨겨지고, 마우스(133)와 연관된 포인터(313)가 스트럿, 또는 그 스트럿에 접속된 콘테이너들, 에지들 또는 가이드들의 근처에 존재하는 경우에 나타난다. 근처는 임의의 수의 스트럿들을 통해 직접적으로 또는 간접적으로 스트럿에 접속되는 임의의 다른 스트럿, 콘테이너, 에지 또는 가이드를 포함할 수 있다.

스트럿이 단지 1 차원의 길이를 갖기 때문에, 1 차원에서의 스트럿 위치는 스트럿이 접속되는 에지들의 현재 위치들로부터 얻어질 수 있다. 그러므로, 스트럿(412)이 에지들(410 및 411) 사이에 그려진다. 수직의 차원에서의 스트럿의 위치는 여러가지 방식들로 계산될 수 있다. 한가지 접근법은 접속된 에지들의 중점들을 평균하고, 평균에 스트럿의 중점을 위치시키는 것이다. 그러므로, 스트럿(412)의 중점은 에지들(410 및 411)의 중점들 사이의 정확히 중간지점이다.

컨테이너의 콘텐츠는 문서 간에 달라질 수 있으므로 콘테이너의 비고정 에지의 위치는 각 문서의 페이지에 따라 달라질 수 있다. 스트럿 제한의 작용은 에지 또는 가이드들 사이의 거리를 유지하므로, 스트럿에 접속된 에지나 가이드의 가능 위치의 범위는 문서 내로 제한할 수 있다.

도 4를 참조하면, 콘테이너(407)의 제1 에지(410)가 이동하는 경우, 예를 들어 사용자가 포인팅 디바이스(313)를 사용하여 사용자 인터페이스를 통해 하나의 에지를 이동하는 경우, 제2 에지(411)는 이러한 두 에지 간의 스트럿 제한(412)을 유

지하기 위해 '밀리거나(pushed)' 또는 '당겨질(pulled)' 수 있다. 사용자는 접속된 에지나 가이드를 밀거나 당김으로써 스트럿이 동작하는 것으로 이해할 수 있지만, 사용자 인터페이스에서의 실제 동작은 문서들을 생성하는데 사용되는 것과 동일하다.

또 다른 구현에서는, 스트럿 툴이 활성화된 동안, 스트럿들이 드러나는 앞서 기술된 방법은, 모든 스트럿 툴이 활성화 될 때마다 모든 스트럿이 드러나도록 변경될 수 있다. 대안적으로, 스트럿들은 포인터가 직접적으로 스트럿의 위에 존재하는 경우에만 나타나게 되고, 어떠한 다른 스트럿들도 동시에 나타나지 않는다.

도 4에 도시된 바와 같이, 스트럿(412)에 의해 접속된 에지들(410 및 411)이 서로에 대해 인접하여 마주보고 있지만, 에지들이 서로 근접하여 마주보는지 아닌지의 여부에 관계없이 분리된 콘테이너들의 임의의 2개의 평행 에지들에는 스트럿들이 적용될 수 있다.

도 11은 콘테이너들간의 스트럿을 도시한 것으로, 도 6a 내지 도 6c를 참조하여 앞서 기재된 콘테이너에 대한 예시적인 규칙들의 제3 세트를 사용하여 도시된다. 제1 콘테이너(1101)는 최소 범위 프레임(1103), 최대 범위 프레임(1105) 및 콘텐츠 프레임(1104)을 갖는다. 스트럿(1106)은 콘텐츠 프레임(1104)의 각각의 에지에 접속된다. 제2 콘테이너(1102)는 최대 및 최소 범위가 동일함을 보여준다. 따라서 콘테이너(1102)는 이 구조에서는 고정된 크기를 갖는다. 스트럿(1106)은 전체 콘테이너(1102)에 작용하고, 그 결과, 수평방향으로 밀리거나 당겨질 수 있다.

따라서, 스트럿들은 제1 콘테이너의 콘텐츠 프레임의 임의의 에지와 최대 범위 프레임 또는 제2 콘테이너의 콘텐츠 프레임의 병렬 에지, 또는 병렬 가이드, 가장자리, 또는 다른 대상 사이에 생성될 수 있다.

10.2 콘테이너들 사이의 스트럿 생성

도 10을 참조하면, 먼저 마우스(133) 및 포인터(313)를 이용하여 툴바 영역(303)에 있는 스트럿 툴 버튼(406)을 클릭하여 스트럿 툴을 활성화함으로써 새로운 스트럿을 생성할 수 있다. 이어서, 두 가지 동작들 중 하나가 두 콘테이너들을 링크하도록 수행될 수 있다. 먼저 제1 콘테이너(1001) 내의 임의의 점(1003)에 마우스(133)를 눌러서 유지한 채로, 포인터(313)를 제2 콘테이너(1002) 내의 임의의 점으로 드래그하여 마우스(133)의 누름 동작을 해제함으로써 경로(1004)가 드래그된다. 이는 도 10의 포인터(313)의 위치에 의해 도시되어 있다. 대안으로, 도 10의 포인터(313)의 위치로 도시된 바와 같이, 콘테이너들(1001 및 1002) 사이의 스트럿은 제1 콘테이너(1001) 내부의 임의의 점(1003)에서 첫번째로 마우스(133)를 클릭(눌렀다 떼기)하고, 이어서 제2 콘테이너(1002) 내부의 임의의 점에서 두번째로 클릭함으로써 형성될 수 있다.

특정 구현에서, 접속될 두 개의 콘테이너들을 선택하는 것뿐만 아니라, 기술한 스트럿 생성 프로세스는, 각각의 콘테이너의 다양한 에지들과 경로(1004)의 시작 및 끝점의 최대 근접성에 기초하여 콘테이너의 접속 에지를 선택한다. 예를 들어, 시작점(1003)은 제1 콘테이너(1001)의 좌측 에지(1005)보다 우측 에지(1006)에 더 가까우므로, 우측 에지(1006)가 접속될 것이다. 유사하게, 제2 콘테이너(1002)의 좌측 에지(1007)가 접속될 것이다.

대안적인 구현예에서, 선택된 에지들은 양쪽 콘테이너들로부터 가장 근접한 평행한 인접 에지들의 가장 가까운 쌍, 예컨대 도 10에 도시된 콘테이너들(1001 및 1002)의 각각의 에지들(1006 및 1007)이 될 수 있다.

또 다른 구현예에서, 선택된 에지들은 제1 콘테이너의 내부로부터 외부로 움직일 때 그 포인터의 경로가 첫번째 가로지른 에지와, 제2 콘테이너의 외부에서 내부로 움직일 때 포인터의 경로가 첫번째 가로지른 에지가 될 수 있다.

사용자가 적절한 제2 에지를 선택하도록 도와주기 위해, 스트럿의 제1 점(1003)이 정의된 후에, 잠재적인 제2 에지들은 포인터가 근사값으로 이동함에 따라 에지의 선 스타일을 변경하거나 에지의 근처에 아이콘을 디스플레이함으로써 그래픽적으로 표시될 수 있다. 제1 점으로 정의된 콘테이너와는 다른 콘테이너들 상에 평행 에지들을 포함하는, 잠재적으로 유효 스트럿을 생성할 수 있는 에지들만이 표시될 것이다.

부가적으로, 스트럿들이 비대칭(non-symmetric)이면, 예를 들어 스트럿이 몇가지 이유 때문에 접속된 에지들 각각을 다르게 처리하면, 기술된 스트럿 생성 처리는 이를 그래픽으로 표시할 수 있다. 예를 들어, 시작점(1003)이 제1 콘테이너(1001) 내부에 있고, 끝점은 제2 콘테이너(1002)의 내부에 존재하므로, 제1 콘테이너의 에지(1006)는 스트럿의 동작에 의해 에지(1007)와 다르게 처리될 수 있고, 스트럿은 그 사실을 그래픽으로 표시할 수 있다.

10.3 가이드들 사이의 스트럿 생성

가이드들 사이에 스트럿들을 생성하는 것은 앞서 기재된 콘테이너들의 에지들간에 스트럿들을 생성하는 것과 유사하다. 가이드들은, 콘테이너들이 그러했던 방식으로 폭과 높이를 갖지 않기 때문에, 더 적은 사용자 인터페이스 만이 가능하다. 다른 접근법은, 스트럿 툴(406)을 선택하고, 이어서 제1 가이드를 선택한 후에, 포인팅 디바이스를 드래그하여 스트럿이 제2 가이드에 부착되어 생성되도록 한다.

또 다른 구현은 두개의 가이드들이 선택되도록 허용하고, 이어서 그 가이드들 사이에 비지향성(non-directional) 스트럿을 생성하는 스트럿 생성 버튼(아이콘)이 선택된다. 순서를 고려하는 다른 구현에 의해 지향성 스트럿이 생성될 수 있으며, 가이드들은 구동한 에지와 구동된 에지를 식별하는 방법으로 선택된다.

10.4 가이드와 콘테이너 사이에 스트럿들 생성

하나의 구현에서, 가이드를 콘테이너의 에지 상으로 드래그함으로써 거리 제한을 생성하고, 통상적으로 이는 가이드와 에지 사이에 0-거리(zero distance)를 지정한다. 이러한 거리 제한은 스트럿에 의해 표현될 수 있으며, 그러므로 가이드를 드래그하는 그러한 방법(가이드 또는 콘테이너의 구성동안 또는 편집동안)은 실제적으로 스트럿을 생성할 수 있다.

스트럿들이 콘테이너 에지들 사이에서 생성되는 방법(툴 선택, 에지 또는 가이드 클릭, 드래그, 놓기)과 유사한 형태로, 스트럿들이 가이드들과 콘테이너 에지들 사이에 바람직하게 생성될 수 있다.

10.5 마진에 스트럿 생성

또한, 페이지 마진과, 가이드 또는 콘테이너 에지들 사이에 거리 제한이 존재하는 것이 바람직할 것이다. 스트럿들은 이러한 목적으로 사용될 수 있다. 일부 구현들에서는, 가이드 및 콘테이너 에지들이 페이지의 에지에 대해 위치가 고정(이는 스트럿과 동일한 효과를 가짐)될 수 있기 때문에, 이러한 것이 필수적이지는 않다. 원하는 페이지 마진에 대해 고정된 가이드가 생성될 수 있고, 다른 가이드 및 콘테이너 에지들이 스트럿을 통해 그에 링크될 수 있으므로, 페이지 마진과 가이드 또는 콘테이너 에지 사이의 스트럿이 또한 불필요하다. 그러므로, 스트럿들이 페이지 마진을 가이드 또는 콘테이너 에지들에 링크할 수 있어야 할 필요성이 존재하지 않는다.

또 다른 구현에서, 가이드들과 콘테이너 에지들은 페이지 마진들에 대해 고정될 수 없다. 이러한 상황에서, 페이지 마진들에 대한 거리 제한을 표현하기 위해 스트럿들을 사용하는 것이 바람직할 수 있다.

10.6 스트럿과 콘테이너들 사이의 상호작용

스트럿들은 다양한 방법들로 상호작용한다. 스트럿의 가장 간단한 동작은 , 콘텐츠가 문서들 내에 배치될 위치를 결정하는 방법으로서, 콘테이너들과 가이드들의 에지들이 위치를 제한하는 것이다. 그러나, 콘테이너로의 스트럿의 접속은, 또한 콘테이너의 상태, 특히 에지 상태를 변경할 수 있는데, 콘테이너는 허용된 상태들의 세트 내에서 유지된다.

특정 구현에서 콘테이너들에 스트럿들의 부착을 통제하는 규칙들은 다음과 같다:

1. 스트럿이 두개의 고정 에지들 간에 접속되면, 선택된 에지들은 비고정된다.

2. 스트럿이 두개의 고정 에지들 간에 접속되면, 제2의 선택된 에지는 비고정되고, 스트럿의 방향에 대응하는 차원에서의 콘테이너의 길이가 고정되면, 스트럿에 수직인 두 에지들은 고정되지 않는 반면, 상기 차원의 콘테이너의 길이는 고정된 채로 유지된다. 예를 들어, 제2의 선택된 에지가 고정된 폭 콘테이너의 수직 에지이면, 콘테이너의 두 수직 에지들이 비고정되어 콘테이너는 폭이 고정된 채로 수평으로 이동할 수 있게 된다.

3. 스트럿에 부착된 콘테이너 에지들 및/또는 가이드들 모두가 고정되면, 스트럿을 제거한다.

4. 고정된 콘테이너 에지 또는 가이드가 사용자에 의해 이동되면, 그에 부착된 임의의 스트럿의 길이 세팅이 변경되어 스트럿에 접속된 다른 에지 또는 가이드가 변경되지 않은 채로 남게된다.

11. 가변 데이터

사용자 인터페이스(103)는, 사용자가 데이터 소스를 가변 문서 템플릿과 연관시키고, 템플릿과 데이터 소스를 통합시키는 경우에 유효 문서들을 생성할 수 있도록 템플릿을 설계하도록 허용한다. 통합은 배경에서 대화식으로 수행되거나, 소프트웨어 애플리케이션(121)의 레이아웃 엔진(105) 구성요소의 요청에 의해 수행된다.

바람직하게 데이터 소스는 가변 문서 템플릿과 대화식으로 연관되고, 레이아웃 엔진(105)은 사용자가 통합된 문서를 대화식으로 네비게이팅하므로 요청에 의해 문서의 페이지들을 통합하고 레이아웃한다. 다른 구현은 데이터를 템플릿과 통합하고, 통합 프로세스동안 대화식으로 사용자를 인터페이스하지 않고 통합된 문서를 프린트할 수 있다. 다른 구현은, 배경에서 데이터를 템플릿과 통합하여 통합 문서를 생성하고, 사용자가 통합 문서를 네비게이팅하는 것과 관련하여 비동기적으로 나타나는 메시지를 통해 통합 프로세스 내의 문제들을 사용자에게 알려줄 수 있다.

11.1 데이터 소스 선택

가변 문서 템플릿을 데이터 소스와 통합하기 위해, 적절한 데이터 소스가 템플릿과 연관될 필요가 있다. 그러한 연관을 달성하기 위한 한가지 방법은 도 13에 도시된 바와 같이, 사용자 인터페이스(103)의 일부를 형성하는, 데이터 소스 선택 윈도우의 이용을 통한 것이다.

도 13은 UI 애플리케이션(103)에 의해 디스플레이(144)에 표현된 GUI 윈도우를 도시하며, 이는 사용자가 데이터 소스를 대화식으로 선택하도록 허용하고, 이러한 데이터 소스는 데이터베이스, 파일, 복수의 데이터베이스들의 결합, 또는 정보의 몇가지 다른 소스가 될 수 있다. 도 14에 도시된 바와 같이 일단 소스가 선택되기만 하면, 소스는 명백하게 또는 암시적으로 연관성이 사라질 때까지 그 템플릿과 연관된다.

11.2 데이터 필터링

일 특정 통합 동작동안 데이터 소스 내의 모든 데이터가 유효하지 않을 수도 있다. 그러한 경우에, 사용자가 사용될 데이터의 범위를 제한할 수 있는 여러가지 가능한 방법들이 존재한다. 하나의 구현은, 알고리즘적 데이터 필터링이 사용될 데이터의 종류와 양을 제한하도록 허용하는 것이다. 예를 들어, 우편번호(postal zip code)의 범위를 선택함으로써, 문서를 생성하기 위해 원하는 지리적 지역에 연관된 레코드만이 사용될 수 있다. 그러한 알고리즘적 필터링은 당해 기술분야에 널리 알려져 있다. 다른 구현은, 데이터 소스가 템플릿과 연관된 후에 사용자가 사용자 인터페이스를 통해 사용되어야할 데이터를 선택하고 표시하도록 할 수 있다.

도 17은, 메뉴 아이템의 선택을 통해 데이터 소스가 가변 문서 템플릿과 연관된 후에 발생하는, 사용자 인터페이스(103)의 GUI(301)를 통해 데이터 필터링 동작을 액세스할 수 있는 방법의 바람직한 접근법을 도시한다.

11.3 데이터 정렬(Data Sorting)

관련 데이터를 찾기 위해 데이터 세트를 시퀀스로 정렬하는 능력은 필터링과 관련된다. 하나의 구현은, 사용자가, 데이터 소스의 사용자-선택 '키(key)' 변수에 기초하여 데이터 소스의 레코드들의 정렬을 선택하도록한다. 모든 레코드들은 선택된 키 변수에서 오름차 알파벳 순서대로 정렬된다. 예를 들어, 키가 변수 "이름"이었다면, 정렬 동작이 수행된 후에, (영문 텍스트의 경우) 이름이 문자 A로 시작하는 기록들이 이름이 문자 B로 시작하는 기록들 전에 위치할 것이다. 적절한 사용자 인터페이스의 일부가 도 15에 도시되어 있는데, 이는 정렬이 데이터 소스를 선택하는 프로세스의 일부로서 발생할 수 있음을 보여준다. 또한, 도 17에 도시된 바와 같이 데이터 소스가 연관된 후에도 메뉴 선택을 통한 정렬이 가능하다.

다른 구현은, 다른 종류의 데이터에 대하여 수치 또는 언어 특정 방법과 같은 사용자 선택가능 정렬 방법을 허용할 수 있다. 다른 구현은, 선택된 키 변수를 저장한 정보의 종류에 기초하여 정렬 방법을 자동으로 선택할 수 있는데, 아마도 이 경우, 자동으로 선택된 방법이 열악한 선택인 경우에는 오버라이드 메카니즘(override mechanism)을 이용할 것이다.

11.4 레코드 네비게이션

데이터 소스는 각각이 관련된 정보의 모음인 레코드들을 포함한다. 예를 들어 하나의 데이터 베이스는 고객을 묘사하는 레코드들을 가지고 있거나, 제품의 레코드들을 가지고 있을 것이다. 데이터베이스 내 레코드들이 순차적으로 정렬되고 사용자가 네비게이트할 수 있으면 유용할 것이다.

바람직한 네비게이션 방법은 심벌들(1805)에 의해 도 18에 도시된 바와 같이, 포인팅 디바이스(313)와 마우스(133)에 의해 활성화될 수 있는 GUI 버튼의 집합을 디스플레이하는 것이다. 각각의 버튼은 레코드들의 시퀀스들을 통해 단계별로 하나씩 전진 또는 후진하거나, 시퀀스의 시작 또는 끝으로의 이동하거나, 또는 일정한 수의 레코드만큼 이동한다. 또한, 하나의 접근법은 포인팅 디바이스(313)/마우스(133) 또는 키보드(132)에 의해 활성화된 메뉴를 포함하고, (도 17에 도시된 바와 같이) 이러한 선택들을 도시하며, 부가적으로 사용자가 특정 번호의 레코드로 네비게이팅하도록 허용한다.

특정 레코드로 네비게이팅하면, 디스플레이 스크린(144) 상에 몇가지 방식으로 현재 레코딩하는 소프트웨어 애플리케이션(121)이 도시된다. 이것은, 해당 레코드를 가변 문서 템플릿과 통합하고, 현재 레코드의 인덱스를 순서대로 디스플레이함으로써 발생할 수 있다.

11.5 변수들

각 데이터 소스는, 그 데이터 소스 내의 레코드들에 적용되는 복수의 이름붙여진 변수(named variable)들로 구성될 수 있다. 예를 들어, 고객 데이터 소스는 고객명과 고객주소에 대한 변수들을 가질 것이다. 이들은 그 변수들의 이름들이 모든 레코드에 대해 동일하더라도, 각각의 레코드에 대한 값들이 다를 수 있으므로 변수들인 것이다.

사용자가 데이터 소스 내의 레코드들을 조사하고, 가변 문서 템플릿과 연관된 변수를 선택하도록 변수들을 디스플레이하는 방법을 갖는 것이 유용하다.

변수를 디스플레이하기 위한 바람직한 방법은, 도 18에 도시된 바와 같이 자유-플로팅(free-floating) 윈도우 내에 수직으로, 이름 및 현재값에 의해 각 변수를 리스팅하는 것이다. 현재 값은 현재 레코드를 검사하고 그 레코드에 대한 데이터 소스에 있는 각각의 변수 값을 찾음으로써 결정된다. 이전에 개략적으로 설명된 바와 같이, 현재 레코드는 레코드 네비게이션을 이용하여 선택될 수 있다. 모든 값들 또는 변수명들이 모두 디스플레이될 필요는 없음을 유의해야 한다. 너무 많은 정보로 인해 윈도우 내에 들어갈 수 없으면, 사용자가 각각의 변수 또는 값을 확인하도록 하는 네비게이션 메카니즘들이 존재할 수 있다. 다른 구현은, 또 다른 윈도우의 일부(다시 말해서, 서브-윈도우)인 유사한 디스플레이를 사용할 수 있다. 문서 템플릿과 연관된 어떠한 데이터 소스도 없다면, 그 윈도우는 도 16에서 도시된 바와 같이 변수들이 없을 것이다.

변수들은 몇가지 종류의 데이터를 포함할 수 있다. 변수는 텍스트 값들을 저장할 수 있거나, 또는 이미지를 저장할 수 있다. 이들은 이하의 논의에서 텍스트 변수들 및 이미지 변수들로서 언급된다.

바람직하게는, 데이터 소스의 변수들은 한가지 종류의 값을 가지므로, 변수가 텍스트 또는 이미지를 디스플레이하지만, 모두를 디스플레이 하는 것은 아니다. 다른 구현은 데이터 소스의 변수들이 레코드당 다른 종류의 데이터를 저장하도록 허용한다. 예를 들어, 하나의 레코드에서, 제품 설명 변수는 텍스트 설명을 포함할 수 있는 반면, 동일한 데이터 소스 내의 또 다른 레코드에서 동일한 변수는 이미지 값을 가질 수 있다.

11.6 텍스트 변수들

텍스트 변수는 데이터 소스의 각 레코드 내에 텍스트 값들을 포함한다. 애플리케이션(121)은, 도 18에서 라벨(1801)에 의해 도시된 바와 같이, 데이터 소스의 변수들의 리스트 내에 그 변수의 이름 다음에 구분되는 라벨을 배치함으로써 변수가 텍스트 변수임을 사용자에게 알려준다. 다른 구현은, 텍스트(1802)에 의해 도시된 바와 같이, 변수의 타입을 알려주는 방식과 마찬가지로 변수의 이름 근처에 변수의 값을 디스플레이할 수 있는데, 동일한 상황에서 이미지 데이터도 텍스트 데이터와 동일하게 나타나는 경우가 있을 수 있으므로, 이러한 접근법만으로는 확실하다는 것이 보장되지 않는다. 애플리케이션(121)은 이러한 기술을 이용하지만, 변수 타입들을 식별하는 주요 방법으로서가 아닌, 변수 데이터를 보여주기 위한 보조 역할로서만 사용한다.

11.7 이미지 변수들

이미지 변수는 데이터 소스의 각각의 레코드에 이미지 값들을 포함한다. 그 값들은 압축된 또는 비압축된 포맷의 이미지 데이터이거나, 또는 이미지 리소스를 위치키기 위한 파일명 또는 다른 액세스 방법이 될 수 있다. 도 18에서 라벨(1803)에 의해 도시된 바와 같이, 애플리케이션(121)은 데이터 소스 내의 변수들의 리스트 내의 변수 이름 다음에 구분되는 라벨을 배치하여 변수가 이미지 변수임을 사용자에게 알려준다. 텍스트(1804)에 의해 도시된 바와 같이, 다른 접근법은 변수명 근처에 변수의 값을 디스플레이하는 것인데, (예컨대)이미지 파일 이름이 텍스트 데이터와 혼동될 수 있으므로, 이러한 방법은 확실함을 보장하지 않는다.

11.8 콘테이너들에 드래그 및 드롭

사용자가 데이터 소스의 변수들을 보면서 가변 문서를 구성할 수 있도록 함으로써, 사용자가 어떠한 변수들이 존재하는지를 알 수 있는 방법을 갖는 것은 유용하다. 또한, 주어진 변수로부터 적절한 타입의 콘테이너를 생성할 수 있도록 하는 것도 유용하다.

애플리케이션(121)은 사용자 인터페이스(103)을 통해, 사용자가 마우스(133) 및 포인팅 디바이스(313)을 사용하여 (도 19에 도시된) 변수 목록의 변수들을 직접 조작하도록 허용함으로써 적절한 타입의 콘테이너를 생성할 수 있도록 허용한다. 사용자는 변수를 선택하고(예를 들어 마킹된 사각형 영역(1902) 중에 하나), 포인팅 디바이스(313)을 이용하여 디자인 표면에 드래그하여 선택(1906)을 해제한다. 이것은, 적절한 콘테이너 타입이 생성되어 해당 변수로 초기화 되어야 한다는 신호로 사용된다. 예를 들어, 이러한 방법에 따라 디자인 표면 상으로 드래그된 (라벨(1901)로 표시된) 텍스트 변수는, (텍스트(1903)에 도시된)현재 레코드에 대한 변수값을 디스플레이하는 텍스트 콘테이너를 생성한다. 만약 이미지 변수(1904)가 디자인 표면으로 드래그 되면, 사용자가 설정한 세팅에 따라서 파일명(1905) 및/또는 이미지 자체를 디스플레이, 스케일링, 정리 및 정렬하는 이미지 콘테이너가 생성된다.

이와 같이 생성된 텍스트 콘테이너의 초기 상태는 사용자가 구성가능하지만, 또 다른 접근법은 사용자가 힌트로 선택한 위치를 이용하여 그 페이지에 콘테이너를 설정할 수 있다.

디자인 윈도우 외부 또는 페이지(페이지가 무효인 경우) 외부와 같은 위치는 무효일 수 있으므로, 위치를 콘테이너의 정확한 위치로서 사용하는 것이 불가능할 수도 있고, 따라서 그 위치는 힌트가 될 수 있다. 콘테이너는 상기 위치에 중심을 두거나, 그 위치에 근접하여 콘테이너의 코너 중 하나가 위치하거나, 페이지 내에 완전히 존재하는 경우처럼 다른 제한들을 위반하지 않으면서 상기 위치에 최대한 가까이 위치하거나, 기본의 또는 랜덤한 위치에 배치되어 상기 위치를 무시할 수도 있다. 포인팅 디바이스를 드래그하여 새로이 생성한 오브젝트를 위치시키는 방법은 많은 구현방법이 존재한다.

또한, 애플리케이션(121)은, 디자인 상으로 드래그된 변수를 위한 현재 레코드 콘텐츠의 모든 것을 저장할 수 있는 충분한 크기의 콘테이너를 생성한다. 그러므로 길이가 긴 텍스트는 크기가 큰 텍스트 콘테이너를 생성할 것이다. 또 다른 접근법은 드래그된 변수의 데이터에 대한 중간 크기를 계산하여, 중간 크기를 디스플레이하기에 충분한 크기의 콘테이너를 생성하는 것이다. 다른 접근법들은 이러한 방법의 변형을 사용하여, 가장 큰 레코드의 데이터 또는 가장 작은 것의 데이터를 저장하기에 충분한 크기의 콘테이너들을 생성하거나, 혹은 일부 다른 알고리즘을 사용할 수 있다.

또 다른 접근법은 디자인 공간에 그 변수를 드래그하여 그 변수의 값을 가지고 있는 콘테이너에 대해 하나의 사각형을 없애는 것이다. 다른 접근법은, 변수를 선택하여 우선 드래그 동작을 수행하지 않고, 콘테이너에 대해 하나의 사각형을 스위핑(sweep out)하는 것이다. 또 다른 대안적인 방법은 포인팅 디바이스로 변수를 두 번 클릭하여 디자인 안에서 알고리즘적으로 선택된 위치 또는 임의의 위치에 일치하는 콘테이너를 가지는 것이다.

다른 구현은, 데이터 소스가 디자인과 연관되는 순간에 데이터 소스 내의 모든 변수들이 콘테이너로서 나타나도록 하고, 그 후에 사용자가 필요에 따라 그러한 콘테이너들을 위치시키거나 삭제하도록 하는 것이다. 이런 방법으로 생성된 콘테이너의 초기 위치와 크기는 데이터 소스 내의 데이터 순서와 각 변수 내의 콘텐츠의 상대적 크기에 좌우될 것이다. 이러한 구현에서는, 특정 변수를 나타내는 콘테이너가 이미 존재한다면, 기존 디자인에 데이터 소스를 연관시킨다고 하더라도 변수를 위한 콘테이너가 생성되지 않을 것이다. 동일한 변수를 참조하는 다중 콘테이너의 생성에는 적합하지 않으므로, 이런 방법은 선호되지 않는다.

또한, 특정한 구현은 데이터 소스 변수와 이미 생성된 콘테이너의 링크를 지원한다. 이것은 사용자가 우선 데이터 소스를 연관시키지 않고도 디자인을 생성할 수 있도록 해준다. 이러한 구현은 유연한 워크플로우를 지원하기 위해 데이터 소스의 연관, 콘테이너들의 생성, 콘테이너들과 변수들의 링크를 제공한다.

이러한 접근법은 기존 콘테이너들이 마우스(133) 및 포인터(313) 조합을 사용하여 구현된 드래그 앤 드롭 메카니즘에 의해 자신의 콘텐츠를 변경할 수 있도록 함으로써, 콘테이너로 변수를 드래그하여 콘테이너가 디스플레하는 것을 재할당하거나, 가지고 있던 것에 추가할 수 있다. 예를 들어, 콘테이너가 "Dear"라는 정적 텍스트를 포함하여 생성되고, "First Name"이라는 이름의 변수가 콘테이너에 드래그된 후, "Last Name"이라는 이름의 변수가 콘테이너에 드래그될 수 있다. 이로 인해, 동작의 모드 및 콘테이너의 세팅에 따라, 콘테이너는 이러한 모든 데이터를 디스플레이하거나, 마지막 변수의 값만을 디스플레이 할 수 있다.

11.9 다중 문서 디스플레이 및 네비게이션

문서들은 하나의 문서 템플릿과 데이터 소스로부터의 데이터를 통합하여 생성되므로, 하나의 통합동작으로 많은 문서들이 생성될 수 있다. 이러한 문서를 네비게이트하고 디스플레이하는 방법은 많은 방법이 존재한다.

하나의 구현은 이 문서들을 디스플레이하고 네비게이트하는 여러가지 방법을 포함하고 있으며, 문서 템플릿이 그 문서 템플릿의 제한과 주어진 통합 문서를 위한 데이터 모두의 뷰를 생성해야 하는 요건을 만족하면서 각 데이터 소스의 기록이 통합되어야 하는 '라이브 프루핑(live proofing)'(이하에 기재됨)에 의존한다. 이러한 구현에서는 사용자 세팅 및/또는 생성된 문서의 성격에 따라, 몇몇 가능한 사용자 인터페이스가 존재한다. 이러한 사용자 인터페이스를 요약하면 다음과 같다:

● 각 문서는 워드 프로세스 문서나 스프레드시트와 유사하게 수평 및/또는 수직으로 조직된 하나의 플로우 문서에 디스플레이 되며, 스크롤 메카니즘, 포인팅 디바이스 및/또는 키보드에 의해 네비게이트된다. 하나의 독립된 메카니즘은 도 18의 버튼(1805)과 같이, 데이터 소스 내에서 레코드들을 통해 네비게이트 하는데 사용될 수 있을 것이다.

● 각 레코드가 통합되어, 많은 페이지들을 가진 하나의 문서를 생성할 수 있다. 이러한 페이지들은 수평 및/또는 수직으로 만들어 질 수 있다. 하나의 독립된 메카니즘은 데이터 소스 내의 레코드들을 통해 네비게이트 하는 데 사용할 수 있다.

● 문서 템플릿이 각 레코드와 통합되어 아주 적은 페이지들을 생성할 경우에는, 몇몇 문서들은 수평 및/또는 수직으로 디스플레이될 수 있다. 예를 들어, 각 문서가 길이방향으로 하나의 페이지만 있다면, 모든 문서를 선형적 순서로 배열하여 윈도우 시스템을 이용하여 디스플레이 할 수 있고 문서들을 네비게이트 하기 위해 스크롤 메카니즘을 사용할 수 있다. 이것은 문서 내 그리고 데이터 소스의 기록들 사이 모두에서의 네비게이션을 공간적 네비게이션으로 매핑시킨다.

● 각 문서가 하나 이상의 페이지를 가지는 경우에는, 이들은 수평으로 조직되고, 수직 방향은 이전 기록을 표현하는 문서 아래에 그 다음 문서를 각각 레이아웃 하고, 데이터 소스의 기록을 검색하여, 하나의 차원에서 문서들의 공간을 네비게이션 할 수 있도록 하는데 사용될 수 있다. 수직 및 수평 방향의 역할은 유사한 효과를 유지하면서, 서로 바뀔 수도 있다.

위의 각각의 경우에서, 레코드간, 페이지 및/또는 문서들 간을 네비게이트 하는 버튼, 메뉴 등과 같은 다른 방법이 존재할 수 있다.

하나의 대안적인 구현으로, 문서 템플릿은 통합된 문서들과 독립적으로 디스플레이 될 수 있다. 통합된 문서들은 편집할 수 없는 '프린트 미리보기(print preview)'형식으로만 생성되지만 그 템플릿과 하나의 기록을 통합한 결과만을 디스플레이 한다. 또 다른 대체 구현으로, 문서 템플릿을 보이게 할 수도 있고, 비상호작용적인 미리보기 형식으로 할 수도 있으며, 문서 템플릿과 통합된 문서들의 미리보기를 모두 편집할 수 있는 라이브 프루핑으로 할 수도 있다. 이러한 작업 모드를 다른 방식으로 결합하는 것도 가능하다.

11.10 프루핑(Proofing)

변수 데이터가 사용되는 온-스크린 프린팅 작업에서 얻게 되는 중요한 산출물은, 통합되는 레코드들의 전체 범위를 표현하는 레코드들, 특히, 변수 데이터 레코드들의 집합으로부터 얻어지는 실제적인 극단적 경우를 효과적으로 검색하여 만들어질 수 있으므로, 사용자는 최종적인 통합 문서가 어떻게 보일 것인지 이해할 수 있다.

미리보기와 프루프(proof)는 통합된 문서가 다른 크기의 변수 이미지와 서로 다른 길이의 변수 텍스트에 어떻게 영향을 받는지 보기 위해 다수의 레코드들을 프린트한다. 프린트하는데 가장 유용한 샘플 레코드들은 다음을 포함한다:

● 대부분의 레코드들이 어떻게 프린트할 것인지; 및

● 디스플레이되는 콘텐츠가 적거나 많은 콘텐츠가 어디에 있는지의 레코드들.

더 복잡한 확장 원리는 다음을 포함한다:

● 텍스트가 너무 작게 감소되었거나 최소 크기에 도달되었을 때;

● 임의의 가변 데이터 텍스트가 디스플레이되지 않을 때;

● 이미지의 크기가 너무 많이 스케일업된 때(프린트의 품질이 열악할 수 있음); 및

● 이미지의 종횡비(aspect ratio)가 너무 많이 일그러진 때.

변수 데이터를 다룰 때에는, 레코드의 수나 문서의 버전이 아주 클 수 있으므로 사용자는 자동화된 프루핑 메카니즘을 사용하지 않고, 모든 통합 문서의 최종적인 외형이 만족할 만한지 각 레코드를 수동으로 재검토해야 할 것이다. 예를 들어, 데이터베이스 내에서 단 하나의 레코드만 불만족스러운 외형을 가지는 콘텐츠를 포함할 수 있으며 그 레코드를 찾기는 어려울 것이다. 이 경우는 레코드의 수가 매우 클 때 변수 데이터 작업에 특히 중요한 사실이다.

모든 문서의 최종적인 외관이 수용가능하고 정확히 프린트될 것을 보장하는 가변 데이터 프린팅 작업들의 미리보기와 프루프-프린팅하는 효과적인 방법이 기술된다. 제한된 수의 레코드가 프린트 해야 할 모든 다른 레코드들을 대표하는 온-스크린 미리보기 및/또는 프루프 프린팅 용으로 선택되었다. 이 레코드들은 모든 레코드의 통합 외형을 분석하고 문서의 외형에 가장 중대한 영향을 주는 레코드들을 결정하는 미리 선정된 규칙의 집합을 이용하여 선택된다.

선택적인 미리보기와 프루핑은 극단적인 경우들을 포함하는 출력 결과들의 프로파일을 제공하므로, 사용자가 전체 작업을 프린트하는 경우 당황스럽지 않게 된다. 이러한 레코드들이 가장 짧은 크기와 가장 긴 크기를 포함하여 극단적 경우를 보게 하는 것은 좋으며, 경제적인 접근 방법은 중간 디스플레이 크기, 가장 극단적 경우, 그리고 중간과 가장 극단적 경우들 사이의 레코드 수를 포함하는 "대표적인" 경우의 짧은 리스트를 포함하도록 하는 것으로 일반화될 수 있다.

11.11 라이브 프루핑

라이브 프루핑은 가변 문서 템플릿과 상호작용적 및 요청에 따라 통합된 변수 데이터를 디스플레이하는 과정이다. 도 20은 아이템들(2010 내지 2020)을 갖는 데이터 소스의 레코드(2002)가 프린트된 경우 레코드의 어떤 부분과 닮았는지를 보여주기 위해서 템플릿과 통합된 데이터 소스의 레코드에서 그래픽 사용자 인터페이스(2000)를 도시한다. 여분의 경계들(2004) 및 색상들(2006)은 가변 문서 템플릿 내에서 정의된 제한을 볼 수 있도록 하기 위해 첨가 되었지만, 이러한 인위적 조작은 프린트를 하기 위한 것은 아니다.

템플릿의 디자인에서 나타나는 에러가 상호작용적인 네비게이션을 통해 발견되지 않을 가능성이 많기 때문에 라이브 프루핑은 독립적 통합 과정보다 장점을 가지고 있다.

애플리케이션(121)의 특정한 구현에서 라이브 프루핑을 사용한다. 대체 구현에서는 라이브 프루핑을 선택으로 사용하거나 전혀 사용하지 않는다.

11.12 선택적 프루핑

선택적인 프루핑은 평균적인 문서의 전형이나, 최소한 평균적인 문서를 따르는 변수 데이터 프린팅 애플리케이션(121)에서 선택된 레코드를 보는 하나의 방법이다.

이것은 사용자가 선택적인 교정 과정에서 측정할 많은 종류의 아이템을 규정할 수 있을 때 유용하다. 다음의 측정법이 개별 콘테이너, 스트럿 링킹 콘테이너, 페이지들, 데이터 소스 내의 레코드들이나 데이터 소스 내의 변수들에 적용될 수 있을 것이다. 이러한 것들은 모두 선택적 프루핑에 대한 아래 기술에서 '설계 아이템(design item)'으로 언급될 것이다. 용어 '통합된 크기(merged size)'는 데이터 소스로부터 얻은 하나의 레코드가 가변 문서 템플릿과 통합된 후 하나의 디자인 아이템에 대한 크기 측정값을 언급한다. 여러가지 크기 측정 방법은 다음에 기술될 것이다.

선택적인 미리보기나 선택적인 프루핑이 활성화되면, 하나의 디자인 아이템이 사용자 상호작용이나 사용자 선호 설정에 의해 선택된다. 모든 레코드는 독립적으로 가변 문서 템플릿과 통합된다. 각 레코드에 대해, 선택된 아이템의 통합된 크기가 기록된다(저장된다). 이러한 통합된 크기는 선택된 디자인 아이템의 통합 크기가 다음과 같은 레코드들을 찾기 위해 비교된다:

- 가장 작은 값;
- 중간, 평균, 또는 몇몇 세트의 중간의 몇몇 다른 측정값; 및
- 가장 큰 값.

가장 관련 있는 레코드를 찾기 위해, 애플리케이션(121)은 문자의 수만이 아니라 각 레코드의 실제적인 디스플레이 크기를 검사한다. 예를 들어, "w"는 "i"보다 더 많은 공간을 점유하고 콘터이너의 크기와 문서의 전체적인 외형에 더 많은 영향을 미친다.

이러한 레코드들이 일단 정의되고 나면, 이들은 사용자에게 온-스크린 미리보기 및/또는 프린팅으로 표현된다.

사용자들은 또한 통합된 문서에서 추가적인 레코드들을 미리보거나 프루프 프린팅 하여 가변 데이터 프린팅 작업에서 모든 레코드들이 정확히 디스플레이 된다는 확신을 높일 수 있다. 일반적으로, 전체 집합의 중앙에서의 편차를 이용해 선택된 디자인 아이템에 대해 가장 다른 통합 크기를 가진 레코드들이 사용자에게 가장 처음 또는 가장 자주 나타나는 순서로 디자인 아이템들을 정렬한다.

11.12.1 최대 및 최소 콘텐츠 크기

하나의 디자인 아이템 크기를 특정하는 유용한 방법은 최대, 최소 폭, 높이, 또는 콘텐츠 면적을 고려하는 것이다. 개별 콘테이너들의 경우, 이러한 측정은 계산이 간단하다. 스트럿 링킹 콘테이너들의 경우, 폭이나 높이는 의미 있는 값이지만, 둘 다 의미있는 것은 아니기 때문에, 면적은 의미 있는 측정이 아니다. 개별 페이지의 경우 콘테이너들의 폭의 합이나 콘테이너들의 높이의 합 또는 콘테이너들의 면적의 합이 사용된다. 레코드들의 경우에는, 그 레코드가 사용될 수 있는 경우 모든 페이지들에 대한 이러한 값들의 합이 디스플레이 될 필요가 있다. 변수의 경우, 전체적 또는 부분적으로 그 변수를 디스플레이하는 모든 콘테이너들이 사용될 수 있다. 선택할 수 있는 다른 접근 방법과 이러한 모든 방법들을 사용하는 것 또한 가능하다.

11.12.2 전형적 크기

콘텐츠의 크기를 측정하는 또 다른 유용한 방법은 주어진 디자인 아이템에 대한 전형적 크기를 고려하는 것이다. 전형적 크기는 평균 폭이나 높이, 또는 평균 면적, 면적들의 합의 평균, 또는 폭과 높이의 제곱의 합의 평균, 또는 다른 가능한 값들이 될 수 있다. 또 다른 방법은 위의 각각의 값들에 대한 평균값 대신 중간값을 이용하는 것이다. 기타 통계적 접근법을 사용할 수 있다.

특정한 아이템의 전형적인 값을 정하기 위해 이러한 방법 중 하나 또는 모두를 사용할 수 있다. 예를 들어, 특정 콘테이너가 평균 면적에 가장 근접한 면적을 가지는 경우에 대한 레코드들이 위치될 수 있다.

11.12.3 여백(white-space) 크기

여백은 얼마나 전형적인 디자인 아이템이 통합된 문서들의 집합 내에 존재하는지에 대한 또다른 유용한 척도이다. 여백은 여러가지 방법으로 정의될 수 있다. 예를 들어, 콘테이너가 특정 레코드와 통합된 경우, 예컨대 콘테이너가 최소 크기를 갖고, 콘텐츠가 그 크기를 채우지 못하는 경우에, 해당 콘테이너 내의 콘텐츠 영역과, 콘테이너의 제한들에 의해 정의된 콘테이너 영역 사이의 영역에 있어서의 차이로서 여백을 정의할 수 있다. 대안으로, 폭이나 높이와 같은 하나의 차원에 있어서, 콘테이너 크기 및 해당 콘테이너의 콘텐츠 크기 사이의 차이로서 여백을 정의할 수 있다.

여백을 최대화하는 것은 콘텐츠 크기를 최소화하는 것과 유사하지만, 동일하지는 않고, 그 반대도 마찬가지다. 최소 크기와 같은 콘테이너 제한이 계산상 역할을 하기 때문에 이 두 가지는 동일하지 않다. 예를 들어, 최대 콘텐츠 면적을 가진 레코드를 찾는 것은 최소 여백 면적을 가진 레코드를 찾는 것과 반드시 같지는 않다.

11.12.4 가장 다른 크기

가변 데이터 프린팅에서 가장 특별히 관심을 가지게 되는 것은 특이한 문서를 위치시키는 경우이다. 이를 위해 특이성에 대한 척도가 요구된다. 이러한 척도는 여러가지 방법으로 정의된다. 이러한 척도의 한 예는 디자인 아이템에 대한 중간 크기로부터의 가장 큰 편차(deviation)를 계산하는 것이다. 크기는 폭, 높이, 면적, 폭과 높이의 합, 폭과 높이의 제곱의 합 등으로 정의된다. 이러한 중간 크기는 데이터 소스 내의 각 레코드에 대해 관심있는 각각의 디자인 아이템을 조사하고, 그 아이템의 크기를 찾아서, 모든 크기의 중간값을 찾아 계산함으로써 결정될 수 있다. 따라서, 각각의 디자인 아이템의 크기와 중간 크기 사이의 가장 큰 차이를 계산할 수 있고, 가장 큰 편차를 갖는 디자인 아이템(들)을 발견할 수 있다.

도 21a와 도 21b는 평균 문서와 각 문서에 대한 콘테이너의 폭과 높이를 이용한 가장 상이한 문서를 계산하기 위한 방법(2100)을 도시한다. 방법(2100)은 애플리케이션(121)의 서브 모듈로서 실행될 수 있다.

절차는 단계(2101)에서 시작된다. 첫째로, 방법(2100)은 데이터 소스 내의 모든 기록을 뒤져서, 2102에서 처음 시작하는 문서에서 각 콘테이너의 위치와 크기를 계산하고 단계들(2103, 2104 및 2105)에 형성된 공정 루프를 이용한다.

단계(2106)에서, 방법(2100)은, 모든 문서들의 폭과 높이를 합하여 문서의 수로 나눔으로써, 문서 템플릿 내의 각 콘테이너의 평균 폭과 높이를 계산한다.

문서 템플릿에서 각 콘테이너에 대해 폭과 높이의 평균값을 알게 되면, 방법(2100)은 모든 기록에 대해 단계(2107)에서 시작해 단계들(2108, 2113 및 2116)에 의해 형성된 루프를 반복한다. 단계들(2108, 2111 및 2115)에 의해 제한된 심화 루프에 의해 정의된 문서 내의 각 콘테이너에 대해, 방법(2100)은 콘테이너의 폭과 단계(2109)에서의 평균 폭간의 차이, 콘테이너의 높이와 단계(2110)에서의 평균 높이의 차이를 계산한다. 단계(2110)은 이 값들의 제곱값을 더하여 그 문서에 대한 해를 정한다. 이 결과에 대한 제곱근은 평균값으로부터 멀어질수록 더 큰 의미를 갖는다.

위에서 해가 계산되면, 주어진 모든 문서에 대한 주어진 모든 콘테이너에 대해, 그 콘테이너가 평균 크기에서 얼마나 근접했는지 알 수 있다. 콘테이너가 가장 낮은 값일 경우, 평균에 가장 가깝다는 뜻이다. 반면에, 콘테이너가 가장 높은 값을 가지면, 평균에서 가장 벗어났다는 뜻이다.

사용자가 가장 평균적인 문서나 가장 특이한 문서가 어떤 것인지를 알려고 하면, 단계(2112)에서 모든 문서의 모든 콘테이너에 대해 각각 이 값이 더해지게 된다. 그 후, 방법(2100)을 마무리짓는 단계(2114)에서 정렬된 리스트가 생성된다.

최저값을 가진 문서는 그 콘테이너들 각각에 대해 평균 문서에 가장 근접한 것이다. 최고값을 가진 문서는 평균 문서와 가장 거리가 멀다. 사용자는 5개의 부적절 문서, 또는 10개의 부적절 문서와 같은 식으로, 요구하는 숫자 만큼의 문서들을 볼 수 있다.

다른 접근법은 콘테이너들의 폭과 높이 이외에 다른 속성에 위의 계산을 수행할 수 있다. 유용하게 사용할 수 있는 다른 속성으로는 콘텐츠의 크기, 폰트 크기(폰트 크기가 커졌다 줄었다 하는 콘테이너의 경우), 문서의 여백 면적, 스트럿의 길이, 또는 콘테이너 에지의 위치 등이 될 수 있다.

11.13 사전(Pre-Flight) 체크

위에서 기술된 크기 측정은 전형, 부전형, 문제성 디자인 아이템들을 발견하는 데 사용할 수 있다. '사전 체크'는 하드 카피 프린트를 하기 전에 통합된 문서의 문제점이나 다른 특성을 발견하기 위해 검색하는 자동화된 사전 프로세스를 설명하기 위해 가변 데이터 프린팅 단계에서 사용되는 용어이다.

이 체크는, 모든 기록들이 만족할 수 있게 프린트 될 것이며 의외성은 없을 것이라는 것을 판단하고 아무런 문제가 없음을 보고하거나, 어떤 형식에서 문제성 있는 기록들에 주의를 주는 등의 체크이다.

바람직하게, 사전 체크는 사용자의 요청으로 수행되고 이 체크에서 아무 문제가 없거나 또는 문제가 발견되었다는 것에 대해 사용자에게 알려준다. 문제가 발견되면, 발견된 최초의 문제가 문제의 성격에 대해 비주얼한 방법과 텍스트 형식의 설명과 함께 사용자에게 디스플레이 된다.

또 다른 구현에서는 문제가 발견되었을 경우 사용자가 이를 조사할 수 있도록 모든 문제를 검색하고 나열할 수 있도록 한다. 이 체크는 사용자가 가변 문서 템플릿을 편집하는 동안 백그라운드에서 수행될 수 있으며, 하나의 윈도우나 서브 윈도우에서 문제들의 리스트를 계속적으로 업데이트 하게 된다. 리스트가 비어 있으면 아무 문제가 발견되지 않았음을 표시하는 내용으로 대체될 수 있다. 다른 경우로, 이 백그라운드 체크는 휴지 상태인 시간에만 일어날 수 있고, 계속적으로 일어날 수도 있고, 다른 시간대 또는 상황에서 일어날 수 있다.

12. 레이아웃 방법 개요

본 개시의 일 태양은 페이지 상에 아이템들을 레이아웃 하는 방법이다. 레이아웃은 레이아웃될 아이템들의 집합의 조합이며, 아이템들이 그 레이아웃에 위치할 위치를 정하는 규칙들이나 제한들의 집합이다. 몇 가지 레이아웃 방법이 다음과 같이 기술되어 있다:

1. 레이아웃들을 정의하는 모델. 레이아웃 모델은 레이아웃에 나타날 수 있는 타입과 아이템들의 속성 및 아이템들이 레이아웃 될 방법을 정의하는 허용된 규칙들이나 제한들을 정의한다. 하나의 데이터 구조는 컴퓨터(101)의 작업 메모리 내에서 레이아웃들을 저장하는데 사용될 수 있다. 몇몇 레이아웃 모델들은 아래에 기술되어 있다.

2. 레이아웃들을 생성하고 편집하는 수단. 이것은 레이아웃들을 생성하는데 사용될 수 있는 동작을 포함한다. 이 동작들은 레이아웃 모델과 일치하는 데이터 구조를 만들기 위해 호출될 수 있는 소프트웨어 기능으로 구현될 수 있다. 이에 대해서는 아래에 기술되어 있다.

3. 레이아웃들 내의 아이템들의 위치와 크기를 계산하는 수단. 아이템들과 규칙들에 의해 정의된 하나의 레이아웃이 주어졌을 때, 이 레이아웃 계산 방법은 아이템들이 규칙들과 일치하게 레이아웃 될 방법을 정하고 컴퓨터(101)에서 실행되는 소프트웨어(105)에 의해 형성될 수 있다. 레이아웃들을 계산하는 수단은 아래에 기술되어 있다.

바람직하게, 특정한 레이아웃 모델과 일치하는 레이아웃들을 생성하고 편집하는 수단과, 그 레이아웃 모델을 위한 아이템들의 위치를 계산하는 수단은 기존에 언급한 레이아웃 엔진(105)이 되며, 소프트웨어의 일부로서 동시에 실행된다. 레이아웃 엔진(105)은 사용자 인터페이스(103)를 포함하지 않음을 주지해야 한다. 각기 다른 많은 사용자 인터페이스는 레이아웃 엔진(105)와 동시에 동작할 것이며, 이에 대해서는 도 1a을 참조하여 이미 기술되었다.

13. 레이아웃 모델 개요

레이아웃 모델들은 기초 레이아웃 모델이며 이 기초 레이아웃 모델에 대한 몇 가지 확장이다. 기초 레이아웃 모델은 레이아웃 아이템들의 기본 속성과, 레이아웃들을 정의하는 데 사용되는 규칙들을 정의한다. 또한 기초 레이아웃 모델을 확장하는데 사용되는 몇 가지 추가적인 규칙들이 정의되었다. 규칙들의 몇 가지 조합들은 간단한 레이아웃 방법을 제시하기 때문에 이점이 있다. 기타 조합들은 사용자가 레이아웃 방법을 이해하기에 간단하기 때문에 이점이 있다.

기초 레이아웃 모델과 추가적 규칙들은 아래에 기술되어 있다. 몇몇 구현들은 이 규칙들의 특별한 조합을 결합한 것으로 설명할 수 있다.

14. 레이아웃 개요의 생성과 편집

통상적으로, 레이아웃은 문서 내의 페이지의 일부와 연관된다. 생성 및 편집 동작들은 초기에 빈 레이아웃으로 생성된 레이아웃 상에 동작하는 것으로 가정되고, 이들은 사용자 인터페이스 소프트웨어(103)에 의해 호출되는 기능들이다. 데이터와 문서 템플릿을 결합함으로써 문서를 생성하는 동안, 사용자에 의한 직접 입력없이 서버 소프트웨어에 의해 동작들이 수행될 수도 있다.

레이아웃을 생성 또는 편집하기 위하여 사용되는 동작들은 바람직하게는, 소프트웨어 사용자에 의해 수행된 동작에 직접적으로 대응되지는 않는다. 사용자가 수행한 하나의 동작은 일반적으로 사용자 인터페이스 소프트웨어(103) 또는 바람직하게는, 소프트웨어의 다른 계층에 의해 다수의 레이아웃 생성 동작으로 변환되는데, 상기 소프트웨어의 다른 계층은 레이아웃 엔진(105)에 의해 직접적으로 지원되는 동작보다 고수준의 인터페이스를 레이아웃 엔진(105)에 제공한다.

레이아웃을 생성 및 편집하는 수단은 최소한 다음을 위한 동작을 포함한다:

1. 아이템 추가;

2. 아이템 제거;

3. 규칙들 추가; 및

4. 규칙들 제거.

다른 동작은 아이템들이 규칙들을 수정하기 위해 추가될 수 있다. 동작들을 편집하는 정확한 형식은 사용되는 특정 레이아웃 모델에 따른다. 많은 다른 형식이 가능하다. 이러한 동작들은 아래에 기술되어 있다.

하나의 레이아웃에 아이템들과 규칙들을 추가할 때, 모든 규칙들을 만족시키는 아이템들을 설정하는 것은 불가능하다는 관점에서 모순된 규칙들의 조합을 추가하는 것이 용이하다. 이러한 레이아웃을 과잉-제한(over-constrained) 되었다고 한다. 반대로, 규칙들에 대한 하나의 주어진 조합은 여러 가능한 솔루션에 적용할 수도 있다. 예를 들어, 모든 규칙들을 만족시키는 아이템들을 찾을 수 있다. 이러한 레이아웃은 과소-제한(under-constrained) 되었다고 한다. 모든 레이아웃이 정확히 하나의 솔루션을 가지는 것이 바람직하다. 따라서, 하나의 레이아웃을 생성하는 데 사용하는 동작 외에, 애플리케이션(121)은 또한 정확히 하나의 솔루션만을 존재함을 확인하는 수단을 결합한다. 이것은 레이아웃 생성 수단과 레이아웃들의 계산 수단에 포함된다.

사용자가 솔루션이 없거나 하나 이상의 솔루션을 가지는 레이아웃을 생성하지 못하도록 하는 것이 바람직하기 때문에, 편집 조작과 함께 일관성과 유일성 체크가 수행될 수 있다. 이러한 체크는 그 조작이 가능한 경우 테스트할 개별 편집 조작 후에 수행된다. 만약 가능하지 않은 조작인 경우, 변경은 불가능하고 사용자에게 적절한 피드백이 즉시 주어져야 한다.

그러므로 일관성 및/또는 유일성 체크는 편집 동작의 일부로서 포함될 수 있다. 대체 방법으로, 레이아웃 계산은 하나의 잘 정의된 솔루션이 레이아웃 생성 및 편집 조작을 이용하여 생성될 수 있는 가능한 각각의 레이아웃에 적용되는지 확인할 수 있다.

레이아웃 솔루션들의 존재와 유일성을 확인하는 방법은 아래에 기술되어 있다.

15. 레이아웃 계산 개요

각각의 레이아웃 모델에 대해, 레이아웃 아이템들의 위치와 크기를 계산하는 적절한 방법이 존재한다. 이 위치와 크기는 이 문제에 용이한 2차원 좌표계에서 정의될 수 있다.

몇 가지 방법이 레이아웃 아이템들의 위치와 크기를 계산하는데 사용된다. 이 방법에는 다음과 같은 것이 있다:

1. 트리 횡단 법(tree traversal method);

2. 단순법;

3. 이차 목적 함수를 갖는 수정된 단순 방법

4. 그래픽 기반 레이아웃.

트리 횡단 법은 기초 레이아웃 모델과 별 차이가 없는 제한된 레이아웃 모델에 적용된다. 다른 방법들은 기본 모델의 다양한 확장에 사용된다. 방법들(2, 3, 4)은 다양한 최적화 문제를 해결하는 것과 같다.

최적화 문제는 최소화 또는 최대화 되어야 하는 제한들과 목적함수의 집합으로 구성된다. 애플리케이션(121)에서, 레이아웃을 정의하는 일부의 규칙들은 제한들을 표현하고, 일부의 규칙들은 목적 함수를 정하는데 사용된다.

단순법은 특별한 계층의 최적화 문제를 해결하는 잘 알려진 방법이다. 단순법은 이 공개문서의 일부로서 기술된 몇몇 레이아웃 모델들과 사용하는 데 적합하다.

이 단순법은 선형 목적 함수만을 지원하므로, 그 레이아웃이 유일한 솔루션을 가지고 있다는 것을 확인하기 위해 고정적 볼록 함수를 가지는 것이 바람직하다. 일반적으로, 이차 목적 함수는 선형 제한과 이차 목적 함수를 가진 최적화 문제를 해결하는 데는 잘 알려진 기법이 있기 때문에, 이차 함수가 사용된다. 이 중 가장 간단한 것은 단순 알고리즘의 병형이다. 이 방법은 최적화 문제를 해결하는 기법으로 잘 알려져 있고 현재 공개 문서의 범주에 관계없이 이들 중 하나를 사용할 수 있다.

16. 레이아웃 모델들의 상세 기술

16.1 기초 레이아웃 모델

애플리케이션(121)의 또 하나의 관점은 아이템들의 집합을 레이아웃 하는 방법을 포함한다. 일반적으로, 이 아이템들은 규칙들과 제한들을 집합과 일치하는 사각형 공간에 레이아웃된다. 도 22는 아이템들과 제한들의 집합을 도시한다.

기초 레이아웃 모델은 이러한 레이아웃들의 정의에 대해 기술되어 있다. 기초 레이아웃 모델은 하나의 레이아웃에 대한 기본 구조를 정의하고 그 레이아웃의 가용성을 제한하다. 예를 들어, 기초 레이아웃 모델은 레이아웃 아이템들을 정의하고, 아이템들의 최대 및 최소 크기와 아이템들의 위치를 정하는데 사용된다. 이 기초 레이아웃 모델에 대한 확장은 레이아웃들 전체에 대한 세부적인 제어를 제공하는 데 사용된다. 기초 레이아웃 모델이 가진 이점은 다음과 같다:

1. 강체와 동체와 같은 물리적 분석 용어로 레이아웃 디자이너에 이해가 용이하고;

2. 그래픽 사용자 인터페이스를 통해 표현 및 처리가 용이할 수 있고;

3. 잘 정의된 동작 세트를 이용해 처리할 수 있는 소프트웨어에서 간단히 표현된다.

도 22에 도시된 바와 같이, 기초 레이아웃 모델에서, 하나의 레이아웃은 박스라고 하는 하나 이상의 사각형 레이아웃 아이템(2201)을 포함하는데, 박스란 수평 및 수직 측면들(2215 및 2216)과 0 이상의 규칙들(2207 및 2211)에 의해 정의된, 박스들을 배치하기 위한 레이아웃 사각형이다.

기본 레이아웃 모델을 갖는 일부 애플리케이션에서는 레이아웃 사각형이 존재하지 않을 수도 있지만, 통상적으로, 아이템들이 레이아웃 되는 사각형 공간은 페이지의 프린트 가능 영역 또는 문서 내의 페이지의 일부를 나타내고, 각 아이템은 그래픽 이미지나 텍스트 블록을 나타낸다.

개별 박스의 크기와 위치는 경계선 사각형의 4개의 모서리의 위치에 의해 정의된다. 크기와 위치는 일반적으로 밀리미터와 같은 절대적인 측정 단위에 관련된 레이아웃 단위에 규정될 수 있다. 규칙들(2207, 2209, 2211, 2213)은 아이템들의 크기들 간의 관계나 레이아웃 사각형을 정의한다. 하나의 규칙은 규칙들(2211, 2213)과 같은 동일한 아이템의 반대 측면들 사이의 관계, 하나의 아이템의 측면과 레이아웃 사각형(2207)의 사이드 사이의 관계, 또는 규칙(2217)과 같이 레이아웃 사각형의 반대 측면들 사이의 관계를 정의한다. 이 기술에서 하나의 측면란, 하나의 아이템이나 오브젝트의 상, 하, 좌, 우를 의미한다.

기초 레이아웃 모델에서, 각 박스들의 사이드는 "정렬 마크(alignment mark)" 또는 단지 "마크(mark)"라고 불리는 추상적 레이아웃 아이템과 연관된다. 레이아웃 영역의 사이드는 또한 마크들과 연관된다. 따라서, 기초 레이아웃 모델은 2 종류, 즉 박스들 및 마크들의 레이아웃 아이템을 지원한다.

"정렬 마크"라는 용어는 다수의 이미지들이 서로의 상단에 프린트되도록 정렬하기 위하여 프린터에 의해 사용되는 등록 마크들의 추상화 및 일반화된 용어이다. 수평 및 수직의 두 가지 종류의 정렬 마크가 존재한다. 수평 마크는 페이지 또는 레이아웃 상의 수직 위치를 나타내며, 부정의 또는 무한의 길이를 가진 수평 라인으로 생각될 수 있다. 수직 마크는 페이지 상의 수평 위치를 나타내며 개념적으로 부정의 또는 무한의 길이를 가진 수직 라인이다.

정렬 마크는 일반적으로 프린트하기 위한 것이 아니다 - 이들은순수하게 박스들간의 위치와 관계를 정의하기 위해 존재한다. 정렬 마크들은 사용자 인터페이스에 의해 다양한 방법으로(또는 필수적으로) 디스플레이 될 수 있다.

레이아웃은 2차원 좌표계와 연관된다. 선택한 좌표계에서 각 마크의 위치나 좌표를 결정하는 것은 레이아웃 엔진(105)의 작업이다. 수직 마크는 수평 좌표를 가지며 수평 마크는 수직 좌표를 가진다. 좌표계는 하나의 지정된 수직 마크와 하나의 지정된 수평 마크에 좌표를 할당하여 규정되는 경우가 많다. 다른 모든 마크의 좌표는 지정된 마크에 대한 상대적 오프셋들로 결정된다. 레이아웃이 고정 크기라면, 좌표계를 규정하는 마크들은 2개의 크기를 가진 레이아웃 사각형인 것이 일반적이다.

박스는 어떤 텍스트와 그래픽과 연관된 사각형 영역이다. 박스는 레이아웃을 계산할 때 하나의 콘테이너를 표현하는데 사용할 수 있다. 각 박스는 박스의 사이드들을 정의하는 4개의 마크와 연관된다. 하나의 박스는 본질적으로 4개의 마크들 간의 관계이다. 기초 레이아웃 모델을 확장했을 때 추가적인 마크가 박스들과 연관될 수 있다. 기초 레이아웃 모델에서, 레이아웃 규칙들이 위치를 결정하는데 사용될 수 있으며 박스의 크기는 사이드들과 마크들의 용어로 기술되지만, 그 모델의 확장에서는, 추가적인 규칙들이 그 박스들과 연관된다.

박스들과 마크들은, 문서에 나타난 개체의 모양 및 위치에 직접적으로 일치하지 않는 모양 및 위치를 표현하기 위해 사용될 수도 있다. 예를 들어, 박스들은, 레이아웃 엔진(105)에 의해 계산된 크기와 다를 수도 있는 아이템들의 이상적 크기를 표현하기 위해 사용된다. 마크들은 다른 아이템들의 위치에 대한 제한을 정의하기 위해 사용될 수 있는데, 예컨대, 마크들은 페이지 마진을 나타내는 위치에 배치될 수 있고, 레이아웃 엔진(105)이 페이지 마진 외부에 아이템들을 배치하지 않는다는 점을 확인하기 위해 규칙들이 추가될 수 있다.

기본 모델에서, 개별 규칙은 수직 오프셋(2209)나 수평 오프셋(2213) 중 하나를 표현하고 특정 크기(2207)이나 음이 아닌 이미지의 크기(2213)(실선으로 도시) 중 하나를 가질 것이다. 고정 오프렛 규칙은 레이아웃 내의 2개의 마크 사이의 수직 및 수평 오프셋의 크기와 방향을 규정한다. 미지의 오프셋 제한은 2개의 마크 사이의 오프셋의 방향만을 규정한다. 예를 들어, 오프셋(2211)은 박스(2219)의 에지(2221)는 반드시 동일 박스의 에지(2223) 위에 있어야 한다는 것을 표시한다. 미지의 오프셋 제한의 크기는 레이아웃 방법으로 계산되어야 하는 값을 표현한다.

그러므로, 기본 모델에서, 고정 오프셋 규칙들과 음이 아닌 오프렛 규칙들의 두 타입의 규칙이 존재한다. 각 타입의 오프셋 규칙은 한 쌍의 마크들의 상대적 위치에 대한 제한을 정의한다.

고정 오프셋 규칙은 1번 마크와 2번 마크 사이의 오프셋은 특정한 값을 가져야 한다는 것을 표시한다. 예를 들어, 주어진 마크 m과 *n*에서, 고정 오프셋 규칙은 fixed(*m*,*n*,d)의 식으로 기술되는 제한이다: fixed(*m*,*n*,d)은 다음 방정식으로 정의된다:

pos(*n*)-pos(*m*) = d,

여기서 d는 음이 아닌 수이다. 스트럿(412)는 레이아웃 엔진에서 고정 오프셋 규칙로 표현된다. 또한 하나의 콘테이너가 고정된 폭을 가지면, 콘테이너의 좌우 사이드를 표현하는 마크들과 관련된 고정 오프셋 제한을 더하여 레이아웃 엔진에 표시될 수 있다. 이와 유사하게, 콘테이너가 고정된 높이를 가지면, 이것은 레이아웃 엔진에서 콘테이너의 상하를 나타내는 마크들 간에 하나의 고정 오프셋 제한으로 표현될 수 있다.

거리 d는 하나의 마크에서 다른 마크까지의 오프셋을, pos(*u*)는 마크 *u*의 위치를 나타낸다. 수평 마크의 위치는 레이아웃 좌표계의 원점에서 그 마크까지의 그 마크의 수직 거리이다. 수직 마크의 위치는 레이아웃 좌표계의 원점에서 그 마크까지의 그 마크의 수직 거리이다. 그 레이아웃을 정의하는 데 편리한 모든 좌표계가 사용될 수 있다. 하나의 고정된 오프셋 규칙에서, 2 개의 마크는 반드시 같은 방향을 가져야 한다. 이런 관점에서, 그 마크들은 둘 다 수평 마크 이어야 하거나 둘 다 수직 마크 이어야 한다.

용어 "고정된"은 이 문서에서 레이아웃 계산 방법에 입력되는 값을 참조하며, 레이아웃 엔진(105)에 의해 변경되지 않는다는 것을 주지해야 한다.

음이 아닌 오프셋 규칙은 2개의 마크 사이의 오프셋이 음이 아님을 규정하는 제한이다. 마크 *m*과 *n* 사이에 음이 아닌 규칙은 non-negative(*m*,*n*)으로 기술되는 제한이며 다음 방정식으로 정의된다.

$pos(n) \geq pos(m)$.

음이아닌 오프셋 규칙에서, 두개으 마크는 반드시 같은 방향을 가져야 한다. 예를 들어, 그 마크들은 둘 다 수평 마크이거나 둘 다 수직 마크이어야 한다.

기초 레이아웃 모델에서, 수평 규칙들은 수직 제한에 의존하므로, 레이아웃의 문제는 2개의 독립적인 규칙들으로 분리된다. 도 23은 도 22에 도시된 레이아웃에 대응하는 수직 오프셋들을 도시한다. 이 구현에서, 거리들은 논리 단위로 지정된다. 도 23에서, 고정된 오프셋 규칙들이 논리 단위의 오프셋 크기로 라벨링되었다. 이 경우, 간단성을 위해 각각의 논리 단위가 1mm라고 가정한다. 논리 단위는 그 애플리케이션에 대해 임의의 편한 크기가 될 수 있지만, 통상적으로 하나의 논리 단위는 가장 작은 독립적으로 어드레스 가능한 프린터 도트보다 작을 것이다. 도 23에 도시된 수직 오프셋 규칙들은 도 24에 도시된 바와 같이 방향 그래프(2400)의 형태로 표시될 수 있다. 도 24에서, 각각의 점(2401)은 수평 마크를 나타내고, 각각의 화살표들(2403)은 규칙을 나타낸다. 고정된 오프셋 규칙들(예를 들어, 2402)은 실선으로 도시되며 음이 아닌(non-negative) 오프셋 규칙들(예를 들어, 2405)은 파선으로 표시된다. 예를 들어, 도 24에서, 루트 노드(2404)와 터미널 노드(2406) 사이의 화살표(2402)는 도 23으로부터의 고정된 오프셋 규칙(2217)을 나타내며, 도 22의 레이아웃 직사각형의 높이를 정의한다. 또한, 고정된 오프셋 규칙들에 의해 설정된 경계들 내에서 음이 아닌 오프셋 규칙들이 변경할 수 있다. 예를 들어, h1 값의 증가가 h2 값 및 h3 값에서 대응하는 감소를 초래하며, 이 두 값은 h1의 터미널 노드(2408)에 의존한다.

통상적으로, 레이아웃 영역은 고정된 크기이다. 이는 레이아웃 영역의 반대 측들과 연관된 마크들을 고정된 오프셋 규칙들과 접속시킴으로써 지정된다. 일부 응용들에서, 레이아웃 엔진(105)이 레이아웃 영역의 측면들과 레이아웃의 아이템들 간의 관계들에 기초하여 레이아웃 직사각형의 크기를 계산하는 것이 바람직하다. 통상적으로, 프린트 가능한 아이템들을 표시하는 박스들이 레이아웃 영역 내에 있도록 제한됨을 보장하기 위해, 규칙들이 레이아웃에 추가되지만, 몇몇 구현들에서는, 이것이 적용되지 않을 수도 있으며, 그 레이아웃 영역 직사각형은 불필요할 수도 있다. 기초 레이아웃 모델의 몇가지 애플리케이션에서, 그 레이아웃 직사각형은 없을 수 있다.

16.2 기초 레이아웃 모델의 대안적인 표현들

또 다른 구현에서, 기초 레이아웃 모델은 앞서 기재된 것과 다르지만 동일한 방식으로 표현된다. 이 경우, 그 기초 레이아웃 모델은 최소 오프셋 규칙이라는 단지 한가지 타입의 규칙만을 사용하여 표현된다.

최소 오프셋 규칙은 제 1 마크와 제 2 마크 간의 최소 허용 오프셋을 지정한다. 마크 *m*과 마크 *n* 간의 최소 오프셋 규칙은 min(*m*,*n*,d)로 표시되는 제한이며 다음의 부등식으로 정의된다:

$pos(n)-pos(m) \geq d$,

여기서 d는 최소 허용 오프셋을 나타내는 수이다. 이 표현에서, 수 d는 양수, 음수, 또는 0이 될 수 있으며, 최소 오프셋 규칙에 의해 관련된 두개의 마크들은 동일한 방향(즉 둘 모두 수평 마크들이거나 또는 둘 모두 수직 마크들)이 되어야 한다.

또 다른 동일한 구현에서, 최소 오프셋 규칙들 대신에 최대 오프셋 규칙들이 사용될 수 있다. max(*m*,*n*,d)로 표시된 최대 오프셋 규칙은 다음의 부등식으로 정의된다:

$pos(n)-pos(m) \leq d$,

여기서 d는 최대 허용 오프셋을 나타내는 수치이다.

이들 두가지 타입의 규칙들 중 단지 하나만이 필요하다는 것을 알기 위해서, 임의의 최대 오프셋 규칙이 다음의 등식 때문에 동일한 최소 오프셋 규칙으로 대체될 수 있음에 유의하라:

$max(m,n,d) \equiv min(n,m,-d)$

기초 레이아웃 모델의 대안의 표현이 원래 기재된 표현과 동일함을 알기 위해서, 임의의 음이 아닌 오프셋 규칙이 다음의 등식으로부터 알 수 있는 바와 같이 최소 오프셋 규칙의 특별한 케이스임을 유의하라:

nonnegative(m,n)≡min(m,n,0)

그리고, 임의의 고정된 오프셋 규칙이 다음 등식에 의해 알 수 있는 바와 같이 두개의 최소 오프셋 규칙들과 동일하다:

fixed(m,n,d)≡min(m,n,d)와 max(m,n,d)

≡min(m,n,d)와 min(m,n,-d).

역으로, 임의의 최소 오프셋 규칙이 다음의 등식으로 도시된 바와 같이 여분의 마크 t, 고정된 오프셋 규칙, 음이 아닌 오프셋을 이용하여 표현될 수 있다:

min(m,n,d)≡fixed(m,t,d)와 nonnegative(t,n), if d≥0

≡fixed(t,m,-d)와 nonnegative(n,t), if d<0,

여기서 t는 m 대비 n의 허용 위치들의 한도를 표시하는데 사용되는 여분의 마크이다. d=0인 특정 경우에서, t를 m으로 대체할 수 있고 고정된 오프셋 규칙은 없어도 된다.

그러므로, 최소 오프셋, 최대 오프셋, 고정된 오프셋, 음이 아닌 오프셋들이 규정될 수 있도록 하는 많은 등식 표현들이 존재한다. 레이아웃들을 표시하는데 사용된 데이터 구조와 그 데이터 구조를 조작하기 위한 소프트웨어를 간소화하기 때문에, 최소 오프셋 규칙들이 이러한 모든 타입의 규칙들을 표현하는데 사용될 수 있다.

기초 레이아웃 모델과 일치하는 레이아웃들을 표현하는데 방향 그래프들이 바람직하게 사용되고, 각 그래프 점은 마크에 대응하며 각각의 그래프 에지에는 그 에지의 소스 점으로 표시된 마크로부터 그 에지의 목적지 점으로 표시된 마크까지 최소 허용 오프셋을 나타내는 수로 라벨링된다. 그러므로, 임의의 주어진 레이아웃에 대해, 그래프 표현이 그 레이아웃 내의 각각의 마크가 그 그래프의 대응하는 점으로 표시되는 곳에 생성될 수 있고, 각각의 최소 오프셋 규칙은 대응하는 유도된 에지로 표시된다. 기초 레이아웃 모델에서 각 박스가 그 측면들에 대응하는 마크들로 완전하게 표시되기 때문에 박스들이 그 표시에서 무시될 수 있다.

컴퓨터 메모리에 그래프를 표시하는 다양한 방법들이 존재하며 임의의 적절한 표시가 사용될 수 있다. 일반적으로 디스플레이하고 편집할 목적을 위한 그래프 표현과는 다른 형태로 아이템들과 제한들로 구성된 레이아웃들을 저장하는 것이 편리하므로, 통상적으로 레이아웃은 대화식 애플리케이션에서 항상 그래프 형식으로 직접적으로 표시되는 것은 아닐 것이다. 그 그래프 표현은 그래프의 에지들 및 점들을 숫자 및 필요한 다른 정보로 라벨링하기 위한 수단을 포함할 수 있다.

도 33a 내지 도 33c는 고정된 오프셋 규칙과 음이 아닌 오프셋 규칙만을 사용하고 또한 단지 최소 오프셋 규칙들 만을 사용하여 콘테이너의 최소 높이와 최대 높이를 어떻게 표시하는지를 도시한다. 도 33a에서, 콘테이너(3300)가 최소 허용 높이인 40 단위와 최대 허용 높이인 100 단위로 도시된다. 콘테이너(3300)는 마크(3301)와 연관된 상부 에지 및 마크(3303)과 연관된 하부 에지로 박스에 의해서 레이아웃 엔진(105)에 표시된다. 도 33b의 방향 그래프는 네개의 점들(3305, 3307, 3309 및 3311)을 갖는다. 점(3305)는 마크(3301)를 표시하고, 점(3307)은 마크(3303)을 표시한다. 나머지 두 개 점들(3309 및 3311)은 마크(3301)에 대해 마크(3303)의 허용 이동 범위를 정의하는 여분의 마크를 표시한다. 이 마크들의 위치는 그래프의 실선 에지(3317)로 표시된 고정된 오프셋 규칙들을 사용하여 마크(3301) 대비 고정한다. 그래프의 점선 에지(3319)로 표현되는 두 개의 음이 아닌 오프셋 규칙은 마크(3303)가 두개의 여분의 마크(3311 및 3309) 사이에 위치하도록 제한하기 위해 사용된다. 도 33c의 두 번째 방향 그래프는 두개의 점들(3313 및 3315) 만을 사용하여 동일한 제한을 표시하며, 여기서 점(3313)은 마크(3301)를 나타내고, 점(3315)은 마크(3303)를 나타낸다. 두개의 에지들(3319)은 최소 오프셋 규칙을 표현하는데 사용된다. 점(3313)에서 점(3315)까지의 에지는 최소값이 40인 최소 오프셋 규칙을 나타낸다. 점(3315)에서 점(3313)까지의 에지는 최소 오프셋 -100인 최소 오프셋 규칙을 나타낸다. 이는 최대 값이 100인 점(3313)에서 점(3315)까지의 최대 오프셋 규칙과 동일하다.

16.3 모양 규칙들

가변 데이터 프린팅 애플리케이션들에서, 박스의 폭과 높이 간의 관계를 정의할 수 있으면 보다 편리하다. 기초 레이아웃 모델에는 이것을 하는 어떠한 방법도 없다.

예를 들어, 박스가 레이아웃에 삽입될 이미지에 대응할 수 있으며 이미지의 크기는 사용가능한 공간 내에 맞도록 조정될 필요가 있다. 이 경우, 박스의 종횡비를 지정하고, 레이아웃 엔진(105)을 갖고 레이아웃의 다른 아이템들의 크기들에 기초하여 그 박스의 최대 크기를 결정하는 것이 바람직하다. 또 다른 예로써, 높이와 폭이 지정되지 않은 텍스트 블록의 경계 박스에 박스가 대응할 수 있는데 이 경우 레이아웃 엔진(105)을 사용하여 높이와 폭 모두를 결정해야 한다.

기초 레이아웃 모델을 확장을 통해 박스의 폭과 높이 간의 관계를 정의하는 모양 규칙이라는 추가 규칙을 마련할 수 있는데 이 때 종횡 규칙과 텍스트 규칙이라는 두 가지 모양 규칙이 추가될 수 있다.

종횡 규칙은 박스가 특정 종횡비를 반드시 가져야 함을 규정한다. 텍스트 규칙은 특정 텍스트 블록의 경계 박스로 박스 모양이 형성되어야 함을 특정한다. 텍스트 레이아웃은 전용 모듈에 의해 처리된다. 텍스트 레이아웃의 예시는 본 문서의 이하에서 설명될 것이다.

모양 규칙의 동작은 다음에 보다 자세히 설명하고 있는데 그 이유는 규칙의 정확한 행동이 레이아웃 계산 방법에 따라 달라지기 때문이다. 이 규칙은 아이템의 위치와 차원을 계산하기 위한 방법에 따라서 구현 별로 다른 동작을 보일 수 있다.

16.4 동일 오프셋 규칙

레이아웃을 규정할 때 편하게 사용할 수 있는 또 다른 제한으로는 두 개의 거리가 동일하다고 하는 것이다. 이를 규정할 때는 마크 m,n,s,t에 대하여 equal(m,n,s,t)로 표시되는 동일 오프셋 규칙으로 규정할 수 있으며, 여기서 m과 n은 동일한 방위를 갖고 s와 t도 서로 동일한 방위를 갖는다. 동일 오프셋 규칙 equal(m,n,s,t)는 다음의 등식으로 정의된다.

$$\text{offset}(m,n) = \text{offset}(s,t)$$

이 식에서 offset(a,b)는 제 1 마크 a에서 제 2 마크 b까지의 오프셋이며 다음의 공식이 성립된다.

$$\text{offset}(a,b) = \text{pos}(b)-\text{pos}(a)$$

동일 오프셋 규칙을 추가하여 기초 레이아웃 모델을 확장할 수 있다. 레이아웃 모델에 동일 오프셋 규칙을 포함하면 제한-과잉인 레이아웃을 생성하는 것이 매우 쉬워진다.

그러나 모든 구현에 동일 오프셋 규칙이 지원되진 않는데 그 이유는 동일 오프셋 규칙이 레이아웃 엔진(105)이 선호하는 레이아웃 방법보다 효율성이 낮은 보다 일반적인 레이아웃 방법을 필요로 하기 때문이다. 레이아웃 엔진(105)은 여러 오프셋을 동일하게 하는 대안 방법으로써의 동일 오프셋 규칙 대신에 아래 정의된 최소화 거리 규칙을 포함하는 경우가 많다.

16.5 박스 중심 포함 규칙

기초 레이아웃 모델에서, 박스의 에지만이 레이아웃 규칙에 관련된다. 박스 중심점 간의 관계를 정의할 수 있는 것이 바람직하다. 기초 레이아웃 모델을 확장하면, 각 박스는 항상 두 개의 부가적인 마크와 연관된다. 이러한 마크들은 수직 마크와 수평 마크이며 두 개 모두 박스의 중심점을 통과한다.

동일 오프셋 규칙을 허용하는 모델 상에서 동일 오프셋 규칙을 이용하여 마크를 각 박스의 중심점과 연관시킬 수 있고 그 결과 박스 중심점 포함 규칙을 지원하는 추가 모델이 불필요하게 된다.

16.6 고정된 중심 규칙들

다른 구현 하에서 박스 중심점과 관련한 규칙의 특별한 케이스의 경우 페이지나 레이아웃 영역의 면 대비 각 박스 중심점의 수직 또는 수평 위치를 고정할 수 있을 때가 있다. 이러한 특별한 케이스에는 동일 오프셋 규칙이 필요 없기 때문에 보다 단순한 레이아웃 계산방법이 사용될 수 있다.

고정된 중심 규칙은 박스의 반대 면들에 대응하는 한 쌍의 마크 간의 관계로 간주될 수 있으므로 레이아웃에 박스 중심점을 통과하는 여분의 마크를 추가할 필요가 없다.

16.7 최대화 오프셋 및 최소화 오프셋 규칙

지금까지 설명한 기본 모델 레이아웃 규칙과 추가 규칙은 아이템의 정확한 크기 및 아이템 간의 공간 등의 정확한 제한과 아이템의 위치 및 크기 변동 한도를 규정하는데 유용하게 사용될 수 있다. 현재까지 기술한 규칙을 가지고 제한-과잉이거나 제한-부족인 레이아웃을 쉽게 정의할 수 있다. 레이아웃을 보다 잘 컨트롤할 수 있기 위해선 기본 모델 규칙과 동일 오프셋 및 고정된 중심 규칙 등의 다른 정확한 규칙을 사용하여 제한-부족 레이아웃을 정의하고 아이템의 바람직한 크기 및 위치를 규정하는 추가의 보다 유연한 규칙을 사용하는 것이 좋다. 이러한 용도로 두 가지 추가 규칙이 사용될 수 있는데 최소화 규칙과 최대화 규칙이 그것이다.

여기까지 정의한 규칙은 레이아웃 모델이 반드시 준수해야 하는 제한을 정의하고 있다. 위에 정의한 규칙과 달리 최소화 및 최대화 규칙은 레이아웃 방법 상의 목적 함수를 정의한다.

minimizeoffset(m,n)로 표시되는 최소화 오프셋 규칙은 마크 m에서 마크 n까지의 오프셋이 (즉 최대한 음수 방향으로)최대한 작아야함을 레이아웃 엔진(105)에게 알려준다. 이 규칙의 이러한 동작은 레이아웃 계산방법에 좌우되기 때문에 아래에 자세히 기술하였다.

maximizeoffset(m,n)로 표시되는 최대화 오프셋 규칙은 마크 m에서 마크 n까지의 오프셋이 (즉, 최대한 양수로)최대한 커야함을 레이아웃 엔진(105)에게 알려준다. 이 규칙의 이러한 동작은 레이아웃 계산방법에 좌우되기 때문에 아래에 자세히 정의될 것이다.

최소화 오프셋과 최대화 오프셋 규칙은 일차 목적 함수에 포함하면 유용하기 때문에 단방향 방법을 사용하여 계산에 적합한 레이아웃 모델에 포함할 수도 있다. 만약 일차 목적 함수를 사용할 경우 각 최소화 규칙이나 최대화 규칙은 각각 목적 함수의 일차 항을 결정한다. 이 경우 이 두 가지 규칙 중 한 가지만 필요한데 왜냐하면 규칙 minimizeoffset(*m,n*)이 maximizeoffset(*n,m*)과 같기 때문이다. 그러므로, 이러한 규칙을 포함한 레이아웃 모델의 데이터 표현식이 이러한 규칙 중 한 개만 지원하면 된다. 예를 들어 데이터 구조가 최소화 오프셋 규칙 만을 지원할 수도 있고 각 최대화 오프셋 규칙을 동일한 최소화 오프셋 규칙을 사용하여 표현할 수도 있다.

16.8 최소화 거리 규칙들

최대화 오프셋과 최소화 오프셋을 사용하면 두 개 마크 간의 이상적인 거리가 무한정인 레이아웃이 될 수 있기 때문에 이 규칙을 사용해서 항상 잘 정의된 레이아웃이 나오는 것은 아니다. 이러한 문제를 피하기 위해 최대화 오프셋 규칙과 최소화 오프셋 규칙을 다른 규칙로 대체할 수 있다. 이러한 규칙을 최소화 거리 규칙이라고 하며 거리를 규정하기 위해 minimizedist(*m,n*)으로 표시한다. 예를 들어 두 개 마크 *m*과 *n* 간의 오프셋 절대값은 가능한 작아야 한다. 최소화 거리 규칙은 같은 방향을 갖는 두 개의 마크 사이에만 적용될 수 있다. 그러나 특정 구현의 경우 최소화 거리 규칙 대신에 아래 기술된 것과 같은 바람직한 오프셋 규칙이라는 다른 규칙을 사용하기도 한다.

다음의 등식을 사용하여 고정된 오프셋 규칙, 최소화 오프셋 규칙, 최대화 오프셋 규칙, 음수 외 오프셋 규칙을 근간으로 최소화 거리 규칙 근사값을 구할 수 있음을 유의하라:

minimizedist(*m,n*)≡minimizeoffset(*m,t*) 및

minimizeoffset(*n,t*) 및

nonnegative(*m,t*) 및 nonnegative(*n,t*)

여기서 *t*는 레이아웃에 추가된 여분의 마크이며 다른 다른 규칙과 관련성이 없다. 이 근사값은 정확하지 않을 수 있으며 사용된 레이아웃 계산방법에 따라 결정된다. 특히, 레이아웃 방법을 통해 계산된 항목의 정확한 위치는 선택한 목적 함수에 따라 그 결과가 차이가 난다.

다음의 등식을 사용하여 최소화 거리 규칙을 기초로 하면 최소화 오프셋 규칙도 근사값을 구할 수 있음을 유념한다.

minimizeoffset(*m,n*)≡minimizedist(*t,m*) 및 fixed(*t,n*,d)

이 공식에서 *t*는 레이아웃에 추가된 마크이며 d는 *m*과 *n* 간의 최대 기대 거리보다 훨씬 긴 거리를 나타내는 양수 중 큰 숫자이다. 이 근사값은 정확하지 않을 수 있으며 사용된 레이아웃 계산방법에 따라 결정되지만 고정된 오프셋과 음이 아닌 오프셋 규칙 만을 허용하는 모든 레이아웃 모델에 최소화 거리 규칙 만을 추가함으로써 최대화 오프셋 규칙과 최소화 오프셋 규칙과 유사한 함수를 얻을 수도 있다.

일부 구현의 경우 최소화 거리 규칙에 규칙의 강도를 보여주는 추가 값이 포함될 때가 있다. 이러한 규칙은 minimizeoffset(*m,n*,s)로 표시될 수 있으며 *m*과 *n*은 규칙과 관련된 마크이고 s는 규칙의 강도를 나타내는 양수이다. 이 경우 마크의 위치에 한 개 이상의 최소화 거리 규칙이 영향을 줄 경우 강도가 약한 규칙보다 강한 규칙이 위치에 많은 영향을 준다.

16.9 바람직한 오프셋 규칙들

특정 구현의 경우 최소화 거리 규칙들을 사용하는 대신 "바람직한 오프셋 규칙"이라 불리는 또 다른 타입의 규칙이 사용된다. Preferred(*m,n*,d)로 표시된 바람직한 오프셋 규칙은 마크 *m*에서 마크 *n*까지의 바람직한 오프셋이 d라고 규정한다. 바람직한 오프셋 규칙을 다음의 등식을 이용 최소화 거리 규칙도 표현할 수 있다.

preferred(*m,n*,d)≡fixed(*m,n*,d) 및 minimizedist(*t,n*)

여기서 *t*는 레이아웃에 추가한 마크로 *m*의 위치 대비 *n*의 바람직한 위치이다.

역으로 다음의 등식을 사용하여 바람직한 오프셋 규칙을 근거로 최소화 거리 규칙을 표시할 수도 있다.

minimizedist(*m,n*)≡preferred(*m,n*,0).

그러므로, 모든 기초 레이아웃 모델 확장에 최소화 거리 규칙이나 바람직한 오프셋 규칙을 추가하면 등식이 나온다. 그러나 최소화 거리 규칙 대신 바람직한 오프셋 규칙을 사용하는데 그 이유는 일반적으로 규정된 레이아웃 관계에 필요한 마크와 규칙이 통상적으로 적게 요구되기 때문이다.

바람직한 오프셋 규칙에도 규칙의 강도를 보여주는 추가 값이 포함될 수도 있다. 이러한 규칙은 preferred(*m,n*,d,s)로 표시되는데 *m*과 *n*은 규칙과 관련된 마크이고 d는 *m*에서 *n*까지의 바람직한 오프셋을 보여주는 숫자이며 s는 규칙의 강도를 나타내는 양수이다. 이 경우 마크에 한 개 이상의 바람직한 오프셋 규칙이 영향을 주면 강도가 약한 규칙보다 강한 규칙이 더 많은 영향을 미친다. 강도가 있는 바람직한 오프셋 규칙은 다음의 등식과 같이 강도가 있는 최소화 거리 규칙과 같다:

preferred(m,n,d,s)≡fixed(m,t,d) 및 minimizedis(t,n,s)

여기서 t는 m 위치에 대한 n의 바람직한 위치를 표시하기 위해 레이아웃에 추가된 마크이다.

minimizedist(m,n,s)≡preferred(m,n,0,s)

16.10 바람직한 레이아웃 모델

하나의 원하는 레이아웃 모델이 상기 정의된 다음 타입의 규칙들을 추가하도록 확장된 기초 레이아웃 모델을 포함한다:

1. 수평적 또는 수직적으로 박스의 중심을 고정하기 위한 규칙들

2. 텍스트와 이미지들을 포함하는 박스들을 지원하기 위한 모양 규칙들

3. 바람직한 오프셋 규칙들

이 레이아웃 모델은 모두 공통으로 필요한 레이아웃 특성들을 지원할 수 있을 정도로 충분히 유연한 반면 사용자들이 충분히 이해하기 쉽고, 간단한 데이터 표현을 하기에 충분히 간단하고 레이아웃 아이템들의 위치 및 차원을 계산하기 위한 신속한 방법을 지원할 수 있다.

16.11 예시 레이아웃

도 37a는 도 4에 도시된 레이아웃 예제가 박스들, 마크들, 규칙들을 이용하여 레이아웃 엔진(105)의 일 실시예에서 어떻게 표현될 수 있는지를 도시한다. 레이아웃 영역의 경계는 마크들(3701, 3703, 3705 및 3707)로 표시된다. 통상적으로 레이아웃 영역은 템플릿 내의 페이지의 일부를 나타낸다. 레이아웃 영역의 높이는 고정된 오프셋 규칙(3709)을 이용하여 표시한다. 레이아웃 영역의 폭은 고정된 오프셋 규칙(3711)으로 나타낸다. 도 4에 도시된 두 개의 콘테이너는 도 37a에 도시된 두 개 박스들(3702 및 3704)에 의해 레이아웃 엔진(105) 내에 표현된다. 마크들(3701 및 3705)은 기원 마크(origin mark)로 지정된다.

제1 박스(3702)의 네 개의 에지들은 마크들(3713, 3715, 3717 및 3719)로 표시된다. 박스(3702)의 상단 좌측 코너는 레이아웃 영역의 좌측 면으로부터 150 논리 단위 만큼의 우측 그리고 레이아웃 영역 상단에서 200 단위만큼 아래의 위치에 페이지 상의 고정 위치를 가진다. 이는 두 개의 규칙들(3735 및 3737)로 표시된 것이다. 규칙(3735)은 박스(3702)의 좌측이 레이아웃 영역의 좌측 면에서 150 논리 단위만큼 우측에 위치함을 보장하고, 규칙(3737)은 박스(3702)의 상부가 레이아웃 영역 상부에서 200 논리 단위만큼 아래에 있음을 보장한다. 박스(3743)의 높이와 폭은 고정되어 있지 않으므로, 레이아웃 엔진(105)에 의해 계산된다. 최소 오프셋 규칙(3744)은 박스의 최속 폭이 120 논리 단위임을 나타낸다. 규칙(3742)는 박스의 최소 높이가 100 논리 단위임을 나타낸다. 바람직한 오프셋 규칙(3741)은 박스(3702)의 바람직한 높이가 550 논리 단위임을 논리 엔진(105)에 나타낸다. 또 다른 바람직한 오프셋 규칙(3743)은 박스(3702)의 바람직한 폭도 550 논리 단위임을 레이아웃 엔진(105)에 알려준다. 박스의 바람직한 폭과 높이는 박스가 나타내는 콘테이너의 콘텐츠들로부터 결정된다. 최소 오프셋 규칙(3739)은 박스(3702)의 하부 에지가 마크(3703)으로 표시된 레이아웃 영역의 하단보다 아래에 있으면 안 됨을 레이아웃 엔진(105)에게 알려준다.

제2 박스(3704)의 네 개의 에지들은 마크들(3721, 3723, 3725 및 3727)로 표시된다. 박스(3704)의 상부 및 하부 에지의 위치는 레이아웃 영역에 대하여 고정된다. 상부 에지 위치는 마크(3725)에 의해 표시된다. 상부 에지는 고정된 오프셋 규칙(3731)에 의해 레이아웃 영역의 상부에서 200 논리 단위만큼 아래에 위치하도록 고정된다. 유사하게, 박스의 하부 에지는 규칙(3733)에 의해 레이아웃 영역의 상부에서 750 논리 단위 만큼 아래에 있도록 고정된다.

박스(3704)의 폭은 고정된 오프셋 규칙(3729)에 의해 550 논리 단위에 고정되지만 박스의 우측과 좌측 면의 위치는 반드시 레이아웃 엔진(105)에 의해 계산되어야 한다. 최소 오프셋 규칙(3747)은 마크(3723)로 표시된 박스의 우측 에지가 마크(3707)로 표시된 레이아웃 영역의 우측 에지를 지나서 확장하면 안됨을 레이아웃 엔진에 지시한다.

추가의 최소 오프셋 규칙들(3748, 3749)은 가변 에지가 레이아웃 영역 내에 포함되도록 한다. 예를 들어 이러한 규칙이 불필요하고 일부 구현의 경우 레이아웃을 계산하기 전에 불필요한 규칙을 미리 제거할 수도 있다. 불필요한 규칙을 파악하기 위해선 한시적으로 규칙을 제거하고 푸쉬 동작을 사용하여 영향을 받는 마크를 푸쉬하여 규칙을 위반할 수 있는지를 볼 수 있다. 푸쉬 동작을 했을 때 규칙이 위반될 수 없으면 그 규칙은 불필요한 것이다.

스트럿(412)은 고정된 오프셋 규칙(3745)에 의해 표시된다. 이러한 규칙은 두 개의 박스들(3702, 3704) 간의 거리가 200 논리 단위이어야 함을 레이아웃 엔진(105)에 나타낸다.

도 37b는 도 37a에 도시된 수평 오프셋 규칙들을 나타내는 그래프를 도시한다. 또 다른 유사한 그래프(도시되지 않음)가 수직 오프셋 규칙을 표시하는데 사용될 수 있다. 점(3751)은 레이아웃 영역의 좌측 에지에 대응하는 마크(3705)를 나타낸다. 점(3761)은 레이아웃 영역의 우측 에지에 대응하는 마크(3707)를 나타낸다. 점(3753)은 박스(3702)의 좌측 에지에 대응하는 마크(3713)를 나타낸다. 점(3755)은 박스(3702)의 우측 에지에 대응하는 마크(3715)를 나타낸다. 점(3757)은 박스(3704)의 좌측 박스에 대응하는 마크(3721)를 나타낸다. 점(3759)은 박스(3704)의 우측 에지에 대응하는 마크(3723)를 나타낸다.

고정된 오프셋 규칙(3735)은 그래프 에지들(3767 및 3769)로 표현된 한 쌍의 최소 오프셋 규칙들의 쌍에 의해 표시된다. 최소 오프셋 규칙(3744)은 그래프 에지(3771)로 표시되고, 최대 오프셋 규칙(3746)은 그래프 에지(3773)로 표현된다. 고정된 오프셋 규칙(3745)은 그래프 에지들(3775 및 3777)로 나타낸다. 고정된 오프셋 규칙(3729)은 그래프 에지들(3779 및 3781)로 표현된다. 최소 오프셋 규칙(3747)은 그래프 에지(3783)로 표시된다.

도 37c는 도 37b에 도시된 그래프를 저장하는데 사용된 메모리 구조를 도시한다. 각 그래프 점은, 최소한 점을 나타내는 마크의 위치와 점에서 출발하는 에지의 인접 리스트를 포함한 데이터 구조의 예를 들어 점(3791)으로 표현된다. 인접 리스트에는 포인터(3793)와 같이 목적 점까지의 포인터를 적어도 포함하는 한 개 에지를 각각 표시하는 기록과 함께 소스 점에서 목적 점까지의 최소 허용 오프셋 (예를 들어 3794)이 들어있어야 한다. 인접 리스트 크기가 다르기 때문에 링크 리스트로 저장된다. 인접 기록 간의 링크는 도 37c에 명시적으로 기술되어 있지 않다.

도 37d에 도시된 바와 같이 바람직한 오프셋은 별도의 그래프에 저장된다. 각각의 바람직한 오프셋은 에지(3795)와 같은, 그래프 내의 에지로 표시된다. 바람직한 오프셋 그래프를 저장하기 위한 데이터 구조는 최소 오프셋 규칙을 저장하는데 사용되는 데이터 구조와 유사하다. 두 개 그래프에 대응하는 점은 포인터들에 의해 링크된다(명확히 도시되지 않음). 일부 구현의 경우, 각각의 에지가 역 에지(reverse edge)를 가짐을 보장하기 위하여 각각의 그래프에 여분의 에지를 추가한다. 이는, 그래프를 전방향 또는 역방향으로 효율적으로 트래버스(traverse)하는 것이 가능토록 하기 위함이다.

17. 레이아웃 생성 및 편집에 대한 상세 설명

17.1 레이아웃 생성 및 편집을 위한 기본 동작들

레이아웃을 생성하고 편집하는 가장 간단한 방법은 모든 아이템을 추가하거나 제거할 수 있도록 하고 지원되는 레이아웃 모델이 허용하는 모든 규칙이 추가되거나 제거될 수 있도록 하는 것이다. 이 경우 제거하는 아이템과 관련된 규칙을 자동적으로 제거함으로써 레이아웃으로부터 아이템을 제거할 때 레이아웃 데이터 구조가 일관성이 없어지는 것을 방지하기 때문에 최소한의 일관성 체크만이 필요하게 된다. 또 규칙을 모두 만족하는 아이템의 위치와 차원을 결정하는 것이 가능하다는 보장이 없다는 문제점이 있다.

불필요한 정보가 들어있는 레이아웃을 피하는 것이 좋은데 왜냐하면 이렇게 하면 레이아웃이 복잡하지 않게 되어 사용자가 이해하기 쉬울뿐만 아니라 불필요한 정보를 제거하는 과정에서 사용자가 레이아웃을 보다 잘 이해할 수 있도록 피드백이 제공되기 때문이다. 불필요한 정보는 또한 레이아웃 데이터 구조 크기를 증가시킬 수 있으며 레이아웃 계산 속도를 늦출 수 있다.

기초 레이아웃 모델의 규칙들을 포함하는 레이아웃을 표현하기 위해 기술된 방향 그래프 구조(도 24)는 일부 타입의 불필요한 정보를 허용하지 않는 수단을 자동적으로 제공한다. 그래프는 동일한 소스 및 목적 점을 갖는 하나 이상의 방향 에지를 허용하지 않는다. 이는, 동일한 오프셋에 하나 이상의 최소 오프셋 규칙이 적용되는 경우를 방지한다. 동일한 한 쌍의 마크 간에 하나 이상의 최소 오프셋 규칙이 존재하는 경우(동일한 순서로 즉 동일한 제1 마크와 동일한 제2 마크를 가짐), 가장 음수의 최소 오프셋 값을 가진 규칙만이 필요하므로 규칙 중의 하나가 불필요하게 된다.

최소 오프셋 규칙을 추가하는 작업은 우선적으로 새로이 추가된 규칙을 갖는 동일한 소스와 목적 점으로 이미 표현된 모든 기존 최소 오프셋 규칙이 자동 교체되게 한다. 그 외에 기존 레이아웃에 최소 오프셋을 추가하는 작업은 만약 동일한 오프셋에 이미 더 작은 (즉 더 음수 값에 가까운) 최소 오프셋 값이 있다면 아무 것도 하지 않게 된다.

특정 구현예에서, 기초 레이아웃 모델에 의해 지원되는 모든 타입의 규칙들을 추가 및 제거하기 위한 동작들이 지원된다. 이러한 규칙은 고정된 오프셋 규칙, 음이 아닌 오프셋 규칙, 최대 오프셋 규칙을 포함하는데, 이들은 모두 최소 오프셋 규칙을 사용하여 표현된다. 예를 들어 고정된 오프셋 규칙을 추가하는 작업은 두 개의 대응되는 최소 오프셋 규칙을 추가함으로써 구현된다. 임의의 마크의 위치를 고정시키거나 고정을 해제하기 위한 동작들이 제공된다. 마크의 위치를 고정하는 것은 마크와 동일한 방향의 원래 마크 사이에 고정된 오프셋 규칙을 추가하는 것과 같다. 또한, 임의의 박스의 중심의 수평 또는 수직 위치를 독립적으로 고정하기 위한 동작들이 제공된다. 레이아웃 생성 및 편집 동작의 많은 조합들이 또한 가능하다.

박스 중심의 수평 위치는 여러 가지 방법으로 효율적으로 고정될 수 있다. 예를 들어, 만약 박스의 두 개 수직 면의 수평 위치가 레이아웃 영역에 대해 고정되어 있다면, 박스 중심의 수평 위치와 폭은 완전히 결정되고 따라서 박스 중심의 수평 위치도 고정된다. 이러한 경우에, 박스 중심의 수평 위치가 레이아웃에 대하여 고정되도록하기 위한(즉, 고정된 중신 규칙을 추가하는) 동작은, 상기 규칙이 불필요한 것이므로, 아무런 의미가 없을 수도 있다. 박스 중심의 수직 위치를 고정할 경우도 마찬가지이다.

일 구현예에서, 바람직한 오프셋 규칙 및 최소 거리 규칙을 추가 및 제거하기 위한 동작이 지원된다. 이러한 구현예에서, 동일한 바람직한 오프셋 규칙을 사용하여 최소화 거리 규칙이 표현된다. 바람직한 오프셋 규칙들은 방향 그래프의 형태로 나타낼 수 있는데, 상기 방향 그래프에서, 각각의 점은 마크를 나타내고, 각각의 바람직한 오프셋 규칙은, 바람직한 오프셋 규칙에 의핸 관련된 마크에 대응되는 점들을 연결하는 에지에 의해 표현된다. 방향 그래프는 동일한 소스 및 목적지 점을 갖는 하나의 에지만을 허용하므로, 동일한 두 개의 마크와 연관된 하나 이상의 바람직한 오프셋 규칙을 갖는 레이아웃을 나타내는 것이 부적절하게 보일 수도 있다. 하지만, 바람직한 오프셋들이 강도를 포함하는 특정 구현예에서는, 동일한 두 개의 마크와 관련된 임의의 두 개의 바람직한 오프셋 규칙들이 다음의 등식을 통해 하나의 동등한 바람직한 오프셋 규칙으로 대체될 수 있음이 밝혀진다:

preferred(m,n,d,s) 및 preferred(m,n,d's')≡preferred(m,n,D,S),

여기서

$$D=\frac{sd+s'd'}{s+s'}$$, and

S = s + s'

이 등식은 바람직한 레이아웃 계산을 수행할 때 유효하다.

17.2 예제 기반 레이아웃 생성 및 편집

레이아웃을 생성하고 편집할 때 제한 과잉의 레이아웃을 만드는 것은 이러한 레이아웃이 문서를 작성하는데 유용하지 않기 때문에 피하는 것이 좋다. 상충되는 규칙을 피하는 과정 또한 사용자에게 피드백을 제공하여 생성되는 레이아웃에 대해 사용자가 보다 잘 이해할 수 있도록 한다. 레이아웃 생성동안 일관성이 없는 규칙을 피하면 자동적으로 실수가 방지됨으로써 사용자의 작업량을 줄일 수 있다.

예시에 기초한 편집을 일부 구현한 경우를 사용하여 제한 과잉 레이아웃을 피할 수 있다. 예시에 기초한 편집은 레이아웃 생성과 편집 작업을 항상 모든 제한을 준수하는 레이아웃 예제에 기초하여 수행하도록 한다. 최소화 오프셋 규칙, 최대화 오프셋 규칙, 최소화 거리 규칙, 바람직한 오프셋 규칙이 레이아웃에 제한을 부과하지 않으며 제한-과잉의 레이아웃을 방지하기 위해서 무시해도 된다는 사실을 유의해야 한다. 가변 데이터 프린팅 애플리케이션의 경우 사용자가 문서의 미리보기를 나타내는 예시 레이아웃을 구성함으로써 이렇게 할 수 있다. 미리보기는 실제 데이터를 문서 템플릿과 결합함으로써 기인한 문서들 중 한 개를 디스플레이할 수 있고 또는 그 미리보기는 문서 템플릿에 삽입될 데이터와 단지 유사하기만 한 샘플 데이터를 포함할 수 있고, 또는 대안으로 템플릿으로부터 생성될 실제 문서들에서 예상되는 바와 적절하게 동일할 수 있다.

새로이 생성된 아이템의 위치와 차원은 아이템이 생성되었을 때 우선적으로 규정되어 아이템의 예시 위치를 항상 알 수 있도록 한다. 이러한 값들을 저장하기 위해서 레이아웃을 표현하는 그래프의 각각의 점에 대응 마크의 위치를 표시한다. 이외에 규칙이 아이템의 현재 위치와 차원에 상충되지 않을 경우에만 규칙을 추가할 수도 있다. 예를 들어 대응 마크의 현재 위치가 최소 오프셋 규칙을 준수할 때만 레이아웃에 최소 오프셋 규칙을 추가할 수도 있는 것이다. 유사하게 고정된 오프셋 규칙을 추가하면 오프셋 값이 반드시 규칙에 의해 연관된 마크 간의 오프셋과 같아야 한다. 박스 중심점의 수평 내지 수직 위치가 고정되면 박스의 위치와 차원이 알려지며 규칙 추가작업을 수행한 시점의 박스 중심점의 실제 위치에 기초하여 수평 또는 수직 중심점이 고정된다.

사용자는 인터페이스(103)를 통해서 레이아웃의 변경을 지정할 수 있으며 레이아웃에 규칙을 추가하는 작업을 요구할 수도 있다. 만약 사용자가 지정한 예시 레이아웃이 추가할 규칙을 만족하지 않을 경우 가능하다면 레이아웃이 자동적으로 조정되어 새로이 추가된 규칙이 레이아웃과 일관성을 갖도록 한다. 예를 들어 사용자가 다이알로그 박스를 통해서 또는 박스의 에지를 드래그 해서 고정된 폭의 콘테이너의 폭을 변화시킬 경우 아이템의 위치가 변하고 규칙도 업데이트 된다. 이를 위해선 일부 기존 규칙 제거, 대응 마크 위치 변경, 새로운 규칙 추가 등과 같은 몇 가지 레이아웃 작업이 필요할 수 있다. 예를 들어 마크의 위치를 변경할 때 애플리케이션(121)은 마크의 위치가 변하면 위반할 수 있는 모든 규칙을 먼저 제거하고 마크의 위치를 업데이트하고 마크의 새로운 위치와 맞는 새로운 규칙을 추가한다.

레이아웃을 편집하거나 생성할 때 제한을 위반하지 않고 다른 마크나 레이아웃 대비 한 개의 마크만 또는 한 개 이상의 마크를 이동하는 것이 필요할 때도 있다. 그러나 움직여야 할 마크가 여러 가지 제한에 의해 영향을 받을 경우 꽤 복잡한 작업이 될 수 있다. 레이아웃 엔진(105)은 마크 세트가 제한을 위반하지 않고 수직적 내지 수평적으로 특정 거리만큼 이동하고 필요하면 원래 없던 다른 마크를 이동하여 소기의 목적을 달성할 수 있도록 하는 "푸쉬"라고 불리는 동작을 제공한다.

17.3 푸시 동작

사용자 인터페이스(103)가 제한을 위반하지 않고 수직적 내지 수평적으로 특정 거리만큼 마크 세트를 이동하고 필요하면 원래 없던 다른 마크를 이동하여 소기의 목적을 달성하고자 한다는 인풋을 레이아웃 엔진(105)가 받아서 푸쉬 동작을 실행한다.

푸시 동작은 모두 동일한 방향의 마크 세트 상에 실행되고 푸쉬 그래프라는 그래프로 기술될 수 있다. 마크 조합과 최소 오프셋 규칙, 레이아웃 내 각 마크에 지정된 위치로 정의되는 예시 레이아웃을 고려하여 푸쉬 그래프를 정의할 수 있다. 푸쉬 그래프는 방향 그래프로 각 점이 레이아웃의 각 점에 대응하며 각 에지는 최소 오프셋 규칙에 대응하는데 이 때 실제 오프셋은 최소 오프셋 규칙에 의해 허용되는 최소값과 같다.

레이아웃의 푸쉬 그래프는 방향 그래프로써 레이아웃을 구성하는 마크의 다른 에지를 직접 푸쉬할 수 있는 마크가 무엇인지가 모서리에 표시된 것이다. 첫번째와 제2 마크 간의 오프셋이 이미 최소 허용치인 경우 제1 마크의 좌표 값을 증가하면 첫번째와 제2 마크 사이 최소 오프셋 규칙을 위반하게 되는데 이 경우 제1 마크가 직접 제2 마크를 (양의 방향으로) 푸쉬할 수 있다. 여기서 레이아웃을 생성 또는 편집할 때 모든 기초 모델 규칙들이 동일한 최소 오프셋 규칙들로 대체됨이 가정됨을 유의하라.

마크를 양의 방향으로 이동하면(즉, 마크의 좌표 값을 증가시키면) 푸쉬 그래프의 대응 점에서 도달할 수 있는 점의 모든 마크도 반드시 동일한 거리만큼 이동시켜 최소 오프셋 규칙이 위반되는 것을 방지해야 한다.

마크를 음의 방향으로 이동하면(즉, 마크의 좌표 값을 감소시키면) 푸쉬 그래프에 역관계인 대응 점에서 도달할 수 있는 모든 마크도 반드시 동일한 거리와 방향으로 이동시켜 최소 오프셋 규칙을 위반하지 않도록 해야 한다. 푸쉬 그래프의 역은 에지들의 방향이 리버스되었음을 제외하고 푸쉬 그래프와 동일하다.

방향 그래프에서 도달가능한 점들의 여러가지 방법이 상기를 구현하는데 사용될 수 있다.

푸쉬 동작은 최소 오프셋 규칙을 위반하지 않고 여러 개의 마크를 이동함으로써 가능하다. 이런 마크들의 허용 이동범위를 계산하기 위해 각 최소 오프셋 규칙 별로 "슬랙(slack)"이라 불리는 양이 정의된다. 최소 오프셋 규칙 min(*m*,*n*,d)이 주어지면, 규칙에 대한 슬랙은 offset(*m*,*n*)-d으로 주어진다. 슬랙은 규칙을 위반하지 않고 최대한 마크 *m*이 (양의 방향으로) 움직일 수 있는 최대 거리이며 규칙을 위반하지 않고 최대한 마크 *n*이 음의 방향으로 움직일 수 있는 최대 거리이다(즉 좌표 값이 감소할 수 있는 최대량임). 예시 레이아웃에서 규칙이 위반되지 않으면 최소 오프셋 규칙의 슬랙은 항상 음이 아닌 숫자이다. 푸쉬 그래프의 각각의 에지는 슬랙이 0인 규칙에 대응한다.

일련의 마크에 허용된 최대 양수 방향으로 이동은 모든 최소 오프셋 규칙 min(*m*,*n*,d)의 슬랙 값들 중 최소값이며, *m*은 마크 세트에 포함되지만 *n*은 그렇지 않다. 이는 마크 세트에 들어있지 않은 마크를 이동하지 않고 최소 오프셋 규칙을 위반하지 않고 마크를 양의 방향으로 이동할 수 있는 최대 거리이다. 음의 방향으로 이동할 수 있는 마크 세트의 최대 거리는 min(*n*,*m*,d) 형식의 모든 최소 오프셋 규칙의 슬랙 값 중 최소값과 비슷하게 정의되며 이 경우 *m*은 마크 세트에 포함되어 있지만 *n*은 그렇지 않다. 만약 특정 방향으로 마크 세트의 이동을 제한하는 최소 오프셋 규칙이 없을 경우 최대 거리는 무한대가 된다(즉 마크들이 얼마나 멀리 이동될 수 있는지에 대한 제한이 없다).

푸쉬 동작은 이동될 마크의 현재 세트 및 현재 세트에서 마크를 푸쉬해야 할 잔여 거리인 푸쉬해야 할 거리를 추적한다. 동작의 단계가 수행됨에 따라 마크의 세트 및 푸쉬할 거리가 업데이트된다. 푸쉬 동작은 마크를 이동시켜 하나 또는 이상 증가시킨다. 각각의 증가의 이동 거리는 마크의 현재 세트의 최대 허용 이동으로부터 결정된다. 세트 내의 마크에 의해 푸쉬될 수 있는 부가적인 마크가 각각의 증가 이전에 세트에 추가되어, 상이한 세트의 마크가 각각의 증가로 이동된다.

도 25는 푸쉬 동작의 제1 구현예와 연관된 방법(2500)을 도시한다. 푸쉬 동작은, 레이아웃 원본이 저장되는 단계(2501)에서 시작된다. 푸쉬 동작은 좌표 시스템을 정의한느 마크의 위치를 변경시킬 수 있으므로, 이러한 마크 및 그 위치를 기억하여 후에 단계(2513)에서 좌표 시스템이 복원될 수 있다. 단계(2501) 후에, 푸쉬 동작이 단계(2502)까지 계속되는데, 이 단계에서, 푸쉬할 잔여 거리를 0과 비교한다. 만약 푸쉬할 잔여 거리가 0이면, 수행할 단계가 존재하지 않으므로, 동작은 좌표 시스템의 원본이 저장된 단계(2513)으로 진행된다. 이는, 단계(2501)에서 저장된 원본 마크의 원본 위치를 복원하기 위하여 적절한 값만큼 레이아웃 내의 마크를 이동함으로써 수행된다. 만약 단계(2502)에서 테스트한 푸쉬 할 거리가 0이 아니면, 동작은 단계(2503)로 진행되는데, 이 단계에서는 레이아웃 내의 마크들의 현재 위치에 기초하여 푸쉬 그래프가 업데이트된다. 그 후, 동작은 단계(2505)로 진행된다. 단계(2505)에서, 이동될 마크들의 세트는, 현재 세트 내의 임의의 마크들에 의해 푸쉬 방향으로 푸쉬될 수 있는 임의의 부가적인 마크들을 추가함으로써 업데이트된다. 예를 들어, 푸쉬 그래프(또는 이동하 거리가 음수인 경우, 푸쉬 그래프의 역)에서 현재 세트 내의 임의의 마크에 대응되는 점으로부터 대응하는 점이 도달 가능한 임의의 마크가 마크들의 현재 세트에 추가된다. 동작은, 현재의 증가를 위한 거리가 계산되는 단계(2507)로 진행된다. 증가를 위한 거리는, 잔여 푸쉬 거리 및 주어진 방향으로의 마크들의 현재 세트에 대한 최대 허용 이동 거리 중에 더 작은 것이다. 이 거리는 푸쉬 거리가 양수일 경우 항상 양수이다. 이동 거리를 계산한 후,동작은 세트(2509)에서 계속되는데, 이 단계에서는 현재 세트 내의 모든 마크들이 단계(2507)에서 계산된 거리만큼 이동된다. 단계(2509) 후에, 동작은 단계(2511)로 진행되는데, 이 단계에서는 잔여 푸쉬 거리에서 이동된 거리를 차감함으로써 잔여 푸쉬 거리를 계산한다. 단계(2511) 후에, 동작은 단계(2502)로 반환되는데, 이 단계에서는, 또다른 순환이 필요한지를 결정하기 위하여 단계(2511)에서 계산된 거리를 테스트한다.

도 25에 기술된 푸쉬 동작의 버전은 결코 실패할 수 없지만, 만약 동작 중에 원본 마크를 이동하면 동작의 행위는 직관적인 이해가 불가능해진다. 예를 들어 만약 다수의 마크가 오른쪽으로 푸쉬되어야 한다면, 마크들이 우측으로 이동된느 대신에, 나머지 다른 마크들이 왼쪽을 이동될 수 있다. 이러한 행위는 단계(2513)에 의해 야기되어, 원본을 회복시킨다. 실질적으로, 이러한 행위가 때때로 유용한다. 예컨대, 박스를 10 거리 단위 만큼 넓히기 위하여, 우측 에지가 10 단위만큼 이동될 수 없더라도, 가능하다면 박스의 우측 에지를 오른쪽으로 10 단위 푸쉬하는 것은 박스를 10 단위만큼 더 넓게 만들 것이다. 이러한 경우에, 우측 에지가 우측으로 3 단위만큼만 이동할 수 있고, 좌측 에지가 좌측으로 7 단위 이동할 수 있다면, 푸쉬 동작은 자동적으로 좌측 에지를 좌측으로 이동시킬 것이다. 부수적인 효과로 다른 아이템들도 또한 이동될 수 있다.

원본 마크를 움직이지 않는 푸쉬 동작의 또 다른 버전을 갖는 것도 또한 유용할 것이다. 이는 방법(2600)에 의해 도 26에 기재된다. 푸쉬 동작의 이러한 구현은 불필요하기 때문에 단계들(2501 및 2503)이 생략되는 것을 제외하면 동일한 방식으로 진행되며, 단지 단계(2606)라는 추가 단계가 단계(2505)와 단계(2507) 사이에 삽입된다. 단계(2606)의 테스트를 통해 현재 데이터 세트에 원본 마크가 포함되어 있는지 여부를 확인한다. 만약 원본 마크가 포함된 경우 동작이 중단되며 실패하는데, 그 이유는 원본 마크를 움직이지 않고 요청한 만큼 마크를 푸쉬할 수 없기 때문이다.

도 27a 및 도 27b, 도 27c 및 도 27d, 도 27e 및 도 27f 모두는 푸쉬 동작이 어떻게 작동하는지의 예를 제공한다. 이 도면들은 두개의 콘테이너, 최소 폭 15 단위를 갖는 가변 폭 콘테이너(2701)와 20 단위 폭을 갖는 고정 폭 콘테이너(2704)로 구성된 레이아웃을 도시한다. 콘테이너(2701)는 마크 A 및 B로 표시된 좌측 및 우측 면을 갖는 박스로 표시된다. 콘테이너(2704)는 마크 C 및 D로 표시된 좌측 및 우측 면을 갖는 박스로 표시된다. 이 그림들은 또한 마크 E로 표시된 페이지(2707)의 에지를 도시한다.

도 27a는 푸쉬 동작을 시작하기 전 레이아웃의 초기 상태를 도시한다. 아이템(2701)은 최소 오프셋 규칙(2702)으로 표현된 15 단위의 최소 폭을 갖는다. 아이템(2704)은 고정된 오프셋 규칙(2705)으로 표현된 20 단위의 고정 폭을 갖는다. 두 개의 아이템(2701 및 2704)은 고정된 오프셋 규칙(2703)으로 표시된 길이 6의 레이아웃 단위 스트럿으로 연결된다. 아이템(2704)과 15 단위의 최소 길이를 갖는 마크 E로 표시된 페이지(2707)의 에지 사이에 부가적인 최소 오프셋 규칙(2706)이 존재한다. 이 규칙은 아이템(2704)이 페이지 에지로 15 단위보다 가까워지는 것을 방지한다. 푸쉬 동작이 발생하기 전의 실제 거리는 20 단위다. 마크 A를 우측으로 15단위만큼 이동하는 푸쉬 동작의 단계가 도시되어 있다. 도 27B에 레이아

웃의 푸쉬 그래프(2709)가 도시되어 있다. 그래프(2709)는, 고정된 오프셋 규칙(2703) 때문에 양방향으로 에지에 의해 연결된 마크 B와 마크 C로 표시되어 있는 점들을 도시한다. B와 C로 표시된 점들 또한 고정된 오프셋 규칙(2705) 때문에 양방향으로 접속된다.

도 27c는 루프의 제1 반복 결과를 도시한다. 제1 반복 시, 단계(2503)에서 푸쉬 그래프(2709)가 계산된다. 단계(2505)에서, 마크 A가 다른 마크를 푸쉬하지 않으며 스스로 이동할 수 있음을 푸쉬 그래프로부터 파악할 수 있다. 단계(2507)에서, 최소 오프셋 규칙(2702)을 위반하지 않고 마크 A가 이동할 수 있는 최대 거리가 3 단위임을 알 수 있다. 3 단위가 요청된 거리보다 작으므로, 단계(2509)에서 마크 A는 3단위 이동하고, 그 결과 도 27b에 도시된 레이아웃이 생기게 된다. 단계(2511)에서 계산된 잔여 거리는 이제 12 단위이다. 루프의 제2 반복 시, 단계(2503)에서 도 27d에 도시된 푸쉬 그래프(2711)가 계산된다. 이 그래프에서, 규칙(2702)의 슬랙이 0 이므로, 마크 A가 마크 B를 푸쉬하며, 결과적으로 마크 A가 마크 C 및 D를 푸쉬한다.

도 27e는 마크 B, C, D 모두가 마크 A에 의해 푸쉬되어, 네 개의 모든 마크가 함께 이동되어야 하는 루프의 제2 반복의 결과를 도시한다. 단계(2507)에서, 마크 D와 마크 E로 표시되는 페이지의 에지(2707) 사이의 제2 최소 오프셋 규칙(2706)을 위반하지 않고, 마크 A, B, C 및 D를 최대 거리 5 단위까지 이동할 수 있음이 판단된다. 잔여 거리는 12 단위이지만, 마크 D는 5 단위만 움직일 수 있으므로, 마크 A, B, C 및 D는 각각 5 단위씩 이동된다. 단계(2511)에서, 남은 거리는 7 단위로 계산된다. 루프의 제3 반복 시에, 단계(2503)에서 도 27f에 도시된 푸쉬 그래프(2713)가 계산된다. 이 그래프에서 모든 마크들이 마크 A에 의해 푸쉬된다.

도 26에 도시된 푸쉬 동작의 제2 버전에서, 테스트(2606)는 원본 마크 E가 이동할 마크 세트에 포함됨을 결정하여, 동작이 중지된다. 도 25에 도시된 푸쉬 동작의 제1 버전에서, 루프의 제3 반복은 단계(2509)에서 모든 마크들이 나머지 7 단위만큼 이동되도록 하고, 다음 단계(2511)에서 잔여 거리가 0으로 계산되기 때문에 단계(2502)의 테스트로 인해 프로세스가 단계(2513)으로 진행되는데, 이 단계에서 모든 마크를 7 단위만큼 좌측으로 이동하여 원본 마크 E의 위치를 복원한다. 이러한 예에서, 두 버전의 푸쉬 동작 모두 동일한 결과를 만든다. 이러한 예에서, 마크 A는 요청한 15 단위 대신 총 8 단위만 푸쉬될 수 있다.

17.4 고정된 중심들을 갖는 푸쉬 동작

고정된 중심 규칙들을 포함하는 레이아웃 모델들에서, 푸쉬 동작이 고정된 중심들을 인식하도록 변경될 필요가 있다. 만약 박스의 중심점의 수평적 위치가 고정되어 있다면 박스의 좌측 및 우측 면은 반드시 항상 반대 방향으로 동일한 양만큼 푸쉬되어야 하는데, 이 때 마크들이 서로 "마주보게 되는" 것으로 언급되거나 "마주보는" 마크들로 언급된다. 유사하게, 만약 박스 중심점의 수직 위치가 고정되면 박스의 상부 및 하단 면이 항상 반드시 동일한 양만큼 반대 방향으로 이동해야 한다.

푸쉬 동작이 고정된 중심점을 포함하도록 적응시키기 위해서, 푸쉬할 마크 세트를 두 개의 마크 세트로 대체한다. 한 개 마크 세트, 즉 정방향 세트는 양의 방향으로 푸쉬되고, 다른 세트, 즉 역방향 세트는 음의 방향으로 푸쉬된다. 단계(2505)에서, 마크들은 다음의 두 개의 마크 세트의 각각에 추가 된다. 만약 마크가 두 개 세트 중 한 개 세트에 포함된 마크와 반대되는 경우, 반대 마크에 의해 푸쉬되는 임의의 마크들처럼 다른 세트에 반대 마크를 추가한다.

도 32는, 단계(2505)에서 사용되는 것과 같이, 고정 중심 규칙이 허용된 경우에 어떻게 두 개의 마크 세트를 결정하는지를 보여주는 방법(3200)을 보다 상세하게 도시한다. 도 32에서, 푸쉬할 초기 마크 세트가 수직 마크이며 우측으로 푸쉬된다고 가정하고, 좌표 시스템이 우측으로 증가한다고 가정한다. 다른 방향과 방위로 마크를 푸쉬하는 것도 유사하게 수행된다. 단계(3201)에서, 현재 마크 세트에 의해 푸쉬 되는 모든 마크를(고정된 중심 규칙을 무시하고) 정방향 세트에 추가한다. 마크들의 정방향 세트를 나타내는 점들의 세트로부터 도달 가능한 모든 점들을 찾음으로써, 푸쉬 그래프를 사용하여 단계(3201)를 계산할 수도 있다. 그리고 만약 이 세트에 두 개의 반대 마크가 포함된 경우, 원본 마크가 세트에 추가된다. 단계(3210)의 다음인 단계(3203)에서, 현재 세트 내의 임의의 마크에 반대되는 모든 마크들을 반대 마크들의 세트에 추가한다. 또한, 임의의 두 개의 반대 마크들이 반대 세트에 포함되면, 그 후에 원본 마크는 반대 세트에 추가된다. 원본 마크를 둘 중의 한 세트에 효과적으로 추가함으로써, 단계(2606)에서의 테스트 후에 푸쉬 동작이 종료된다. 단계(3203)의 다음 단계(3205)에서, 반대 세트의 마크에 의해 좌측으로 푸쉬된 모든 마크를 반대 세트에 추가한다. 마크가 음의 방향으로 푸쉬되기 때문에(즉, 좌표가 감소하기 때문에) 푸쉬 그래프의 역이 사용된다는 점을 제외하고, 이 과정은 단계(3201)와 동일한 방식으로 수행된다. 단계(3205)의 다음 단계(3207)에서, 반대 세트의 마크와 반대되는 모든 마크를 두 세트 중 반대 세트 외의 나머지 한 세트에 추가한다. 단계(3207)는 단계(3203)의 역 과정이다. 단계(3207)의 다음 단계(3209)에서, 단계(3207)의 반대 세트 외의 나머지 세트에 새로운 마크가 추가되었는지를 체크하기 위해 테스트를 실시한다. 만약 추가된 경우, 계산이 단계(3201)로 반환되어,두 개 세트 모두에 새로운 마크가 추가되지 않을 때까지 단계(3201, 3203, 3205 및 3207)가 반복될 것이다.

단계(2507)에서 두 개 세트에 각각 별도로 이동거리가 계산되며 두 개 거리 중 적은 것 (즉 크기가 작은 것)이 이동할 거리가 된다. 단계(2509)에서 마크를 동일한 거리만큼 이동하지만 반대 방향으로 이동된다. 한 개 세트의 마크는 양의 방향으로 움직이고 다른 세트의 마크는 음의 방향으로 이동하는 것이다.

17.5 모양 규칙들을 이용한 푸쉬 동작

모양 규칙이 수평 마크와 수직 마크와 관련되기 때문에 푸쉬 동작에 모양 규칙이 있는 레이아웃 모델을 포함하는 것은 잠재적으로 복잡한 일이다. 문제는 마크를 푸쉬할 때 모양 규칙과 다른 규칙이 조합되어 동일한 방위의 마크 간의 복잡한 상호작용이 발생할 수 있다는 점이다. 특히 순환적 의존성을 유발할 수 있는데 여기에는 한 가지 해결방법이 존재하지 않는다는 문제점이 있다. 이러한 문제를 피하기 위해서 모양 규칙이 최소 오프셋 규칙 상에 수직 면에 대응하는 것은 최대 한 개 마크 밖에 없고 수평 측에 대응하는 마크가 기껏해야 한 개밖에 없는 박스에만 적용된다는 제한을 레이아웃 모델에 부과한다. 즉 최소한 모양 규칙 박스의 한 쌍의 반대 면 중 한 개가 최소 오프셋 규칙에 포함되어야 한다는 의미이다. 이러한 제한이 있으면 모양 규칙으로 인해 발생할 수 있는 마크 간의 복잡한 관계를 방지할 수 있다. 이러한 제한을 통해서 단계

(2509)에서 모양 규칙과 관련된 박스의 면이 움직이면 모양 규칙이 적용될 수 있도록 푸쉬 알고리즘을 변환할 수도 있다. 이 때 모양 규칙은 박스의 자유로운 에지를 이동함으로써 적용할 수 있는데 이 때 자유로운 에지는 최소 오프셋 규칙이 부과되지 않는 에지를 뜻한다.

17.6 문서 탬플릿들을 편집하기 위한 푸쉬 동작 사용

GUI(301)는 사용자가 문서 탬플릿을 편집하는 동안 예시 레이아웃을 변경하기 위해 푸쉬 동작을 사용한다. 도 34는 GUI(301)을 이용하여 어떻게 사용자가 콘테이너의 고정된 모서리를 움직일 수 있는지를 보여주는 예시 방법(3400)이다. 단계(3401)에서, 사용자는 예컨대, 마우스(133)와 포인터(313)로 에지를 선택하여 드래그 시킴으로써 에지가 이동해야 한다고 지정한다. 단계(3401)의 다음 단계(3402)에서, 프로그램(103)은 이동한 에지에 대응하는 마크 위치의 고정 상태를 해제한다. 통상적으로 에지를 나타내는 마크 및 에지와 동일 방위를 갖는 원본 마크 간의 오프셋을 고정하는 고정된 오프셋 규칙을 추가함으로써 에지 위치가 고정된다. 에지의 고정 상태를 해제하려면, 에지의 위치를 변하게 할 수 있도록 고정된 오프셋 규칙을 제거해야 한다. 단계(3402)의 다음 단계(3403)에서, 레이아웃 엔진(105)에 추가 제한을 추가하여 에지의 허용 위치의 한도를 정한다. 에지가 고정되면, 에지를 움직일 수 없기 때문에 에지 이동과 관련된 추가 제한을 부과할 필요가 없다. 그러나, 에지의 위치를 변경하기 위해 레이아웃 엔진(105)을 사용하는 경우, 통상적으로 에지의 이동 범위가 제한된다. 예를 들어, 에지가 레이아웃 영역의 에지를 지나서 이동할 수 없음을 레이아웃 엔진(105)에 전달할 수 있다. 추가 예로써, 만약 에지가 콘테이너의 좌측 에지인 경우, 레이아웃 직사각형의 좌측 면을 표시하는 마크 및 콘테이너의 에지를 표현하는 마크 사이에 음수가 아닌 오프셋 제한을 추가할 수 있다. 또 콘테이너의 최소 및 최대 폭이 실시되어야 한다. 문서 탬플릿을 사용하여 레이아웃 엔진(105)과 상관없이 각 콘테이너의 최대, 최소 폭과 최대, 최소 높이를 저장할 수 있다. 콘테이너의 좌측 및 우측 에지들을 표시하는 마크들 사이에 최소 오프셋 제한과 최대 오프셋 제한을 추가함으로써, 사용자가 콘테이너의 에지를 드래그하는 동안 콘테이너의 최소 및 최대 폭 세팅들이 레이아웃 엔진(105)에 의해 실시될 수 있다.

단계(3403)의 다음으로, 단계(3405)에서 애플리케이션(121)은 에지를 움직이기 위해 푸쉬 동작을 유발한다. 푸쉬 동작은 에지를 이동해도 제한을 위반하지 않도록 하며, 레이아웃 규칙이 허용하는 범위 내로 이동을 제한한다. 푸쉬 동작 후 단계(3407)에서는, 단계(3403)에서 추가된 제한이 제거되고, 그 후 단계(3411)에서 에지의 위치가 새로운 위치에 고정된다. 단계(3411) 다음의 단계(3413)에서, 애플리케이션(121)은 레이아웃 엔진(105)을 이용하여 이하에서 기술되고, 도 28, 도 29, 도 30a 및 도 30b에 도시된 것과 같이 레이아웃을 재계산한다. 단계(3413)는 모든 최소 거리 규칙과 바람직한 오프셋 규칙이 적용되도록 보장한다. 단계(3411)는 단계(3413)를 수행할 때 레이아웃 엔진(105)이 모서리를 다른 위치로 이동하는 것을 방지하기 위해 필요하다. 마지막으로, 단계(3413)에서 레이아웃이 재계산된 후에, 애플리케이션(121)이 디스플레이(144)를 업데이트하여 에지 위치 변화 결과를 보여주고 업데이트가 끝나게 된다. 도 34의 단계들은 드래깅동안 즉각적인 피드백을 제공하기 위해 사용자가 마우스로 콘테이너 에지를 드래그하는 것을 반복할 수 있다.

도 35는 문서 탬플릿을 편집하는데 푸쉬 동작이 어떻게 사용될 수 있는지에 대한 또 다른 방법(3500)을 도시한다. 방법(3500)은 단계(3501)에서 시작하며, 이 단계에서 사용자는 고정된 폭 콘테이너의 폭의 변화를 요청한다. 콘테이너의 위치가 가변적이며 좌측 및 우측 면이 고정되지 않았다고 가정한다. GUI(301)가 디스플레이하는 특성 다이얼로그에 키보드(132)를 이용하여 새로운 폭을 타이핑하여 이러한 과정이 수행될 수 있다. 일 실시예에서, 마우스(133)의 우측 버튼을 이용하여 콘테이너를 클릭하면 팔레트(311)와 유사한 콘텍스트 메뉴가 디스플레이 된다. 그 후, 콘텍스트 메뉴에서 "Properties..." 아이콘을 선택하여 특성 다이얼로그가 디스플레이 되도록 한다. 단계(3501) 다음인 단계(3503)에서, 콘테이너의 현재 폭과 새로운 폭 간의 차이가 계산된다. 단계(3503)의 다음 단계(3503)에서, 콘테이너의 폭이 고정에서 고정해제로 바뀐다. 콘테이너의 좌측과 우측 면을 나타내는 마크 간에 고정된 오프셋 규칙을 추가함으로써 콘테이너의 폭을 고정한다. 마찬가지로, 단계(3505)는 고정된 오프셋 규칙을 제거하는 것으로 구성된다. 단계(3505)의 다음 단계(3507)에서, 콘테이너 면의 이동을 제한하기 위해 레이아웃에 제한을 추가한다. 각 콘테이너의 최소 및 최대 폭이 문서 탬플릿에 존재할 수 있다. 레이아웃에 규칙을 추가함으로써 이러한 제한을 적용할 수 있다. 최소값은 콘테이너의 좌측과 우측면들을 표시하는 마크들과 관련된 최소 오프셋 규칙을 추가함으로써 적용되며, 최대값은 콘테이너의 좌측과 우측면들을 표시하는 마크들과 관련된 최대 오프셋 규칙을 추가함으로써 적용된다.

단계(3509)는 단계(3507) 다음에 이어진다. 단계(3509)에서, 폭의 차이만큼 콘테이너의 우측 면에 대응하는 마크를 우측으로 이동시킬 때 푸쉬 동작이 이용된다. 사용자가 요구하는 작은 폭을 요구하는 것을 반영하여 폭의 변화가 마이너스인 경우 마크는 마이너스 거리만큼 푸시된다 - 따라서 우측 대신 좌측으로 푸시되는 것이다. 다음 단계(3511)에서 필요한 나머지 폭의 변동이 계산된다. 단계(3509)의 푸쉬 동작이 계산된 거리만큼 우측 에지를 성공적으로 이동시킨 경우 콘테이너의 폭이 맞는 크기로 변화하며 남은 거리는 0이 된다. 만약 단계(3509)의 푸쉬 동작이 성공하지 못했을 경우 현재 폭은 요청한 폭과 같지 않고 남은 거리는 0이 아닐 것이다. 단계(3511)의 다음 단계(3513)에서 좌측 면을 푸쉬 동작을 통해 단계(3511)에서 계산된 남은 변동만큼 좌측으로 푸쉬 된다. 남은 폭 변화가 음인 경우, 좌측 에지는 음의 거리만큼 좌측으로 푸시될 것이다. 즉, 그 에지는 우측으로 푸쉬될 것이다.

단계(3513)가, 단계(3511)에서 계산된 거리만큼 좌측 에지를 푸시하는데 성공한 경우, 이제 폭은 요청된 폭과 동일할 것이다. 그렇지 않을 경우라도 폭은 최대한 요청한 폭에 근접할 것이다. 단계(3513) 후의 단계(3515)에서, 단계(3507)에서 추가된 제한들이 제거되고, 단계(3517)에서 콘테이너의 좌측과 우측 에지를 표시하는 마크와 관련된 고정된 오프셋 제한을 추가하여 폭이 새로운 폭으로 고정된다. 단계(3517)의 다음 단계(3519)에서, 레이아웃이 재계산되며 단계(3521)에서 새로이 계산된 레이아웃을 반영하기 위해 디스플레이가 업데이트되고 그 결과 콘테이너의 폭을 변동하는 작업이 완료되게 된다. 레이아웃을 계산하는 과정이 아래에 보다 자세하게 기술되어 있다. 도 35의 예시에서 도 26의 두 번째 푸쉬 동작이 사용되었다고 가정한 것이다. 도 25에서 개략적으로 설명된 버전을 사용하면, 단계들(3511 및 3513)은 불필요하게 될 것이다.

18. 레이아웃 계산에 대한 상세 설명

18.1 단순 알고리즘을 이용한 레이아웃 계산

일 실시예에서, 허용된 규칙들은 동일 오프셋 규칙들, 방향 규칙들, 최소 오프셋 규칙들 및 최대 오프셋 규칙들과 기본 모델 규칙이다. 여기서 기본 모델 규칙은 일차 부등식 제한과 같고, 동일 오프셋 규칙은 일차 제한과 동일하며, 일차 목적 함수를 정의하기 위해 최소 오프셋 규칙과 최대 함프셋 규칙이 사용된다. 이러한 모델에서, 규칙들은 일차 프로그램을 정의하므로, 단순 알고리즘 또는 일차 프로그램들을 해결하는 임의의 다른 방법을 사용하여 레이아웃 계산이 수행될 수 있다.

이러한 실시예에서, 최소 오프셋 규칙에 의해 관련된 한 쌍의 마크들 간의 각 오프셋을 더하고 이 합으로부터 최대 오프셋 규칙에 의해 연관된 한 쌍의 마크들 간의 각각의 오프셋을 감산함로써 목적 함수가 계산된다. 만약 규칙들이 강도를 갖는다면, 그 오프셋은 더하거나 빼기 전에 대응하는 규칙의 강도로 먼저 곱해진다.

불행하게도, 단방향 방법에 커다란 변경 없이도 단방향 방법을 사용하여 레이아웃을 계산할 수 있는 모델들에 텍스트 규칙들을 포함하는 공지된 방법이 존재하지 않는다

18.2 변형된 단방향 방법

또 다른 실시예에서, 허용된 규칙들은 동일 오프셋 규칙들, 방향 규칙들, 최소 거리 규칙들, 바람직한 오프셋 규칙들을 더한 기본 모델 규칙이다. 이러한 실시예에서, 최소 오프셋 규칙 및 동일 오프셋 규칙이 일차 제한으로 전환되며, 최소화 거리 규칙 및 바람직한 오프셋 규칙을 사용하여 이차 목적 함수를 정의한다. 이러한 모델에서, 제한은 일차 부등식이나 일차 등식 모두 다 가능하며 목적 함수는 이차 함수이다. 이러한 타입의 문제들은 이차식 최적화 분야의 당업자들에게 공지되어 있는 방법을 이용하면 해결될 수 있다. 이러한 모델에서, 단방향 방법의 변경된 버전이 레이아웃들을 계산하는데 사용될 수 있다.

단방향 방법과 유사하게, 이차 프로그래밍의 상당한 변경없이도 이차 프로그래밍을 이용하여 가변 폭 및 높이를 갖는 텍스트 박스들을 허용하는 모델들을 처리하는 공지된 방식이 존재하지 않는다.

18.3 그래프에 기초한 레이아웃 계산

또 다른 실시예에서, 사용된 레이아웃 모델은 기초 모델 규칙들, 바람직한 오프셋 규칙들 및 모양 규칙들 모두를 허용한다. 이 모델은, 고정된 중심 규칙이 허용되지 않는다는 점을 제외하고 바람직한 레이아웃 모델과 동일하다. 이러한 실시예에서, 기본 모델 규칙을 최소 오프셋 규칙을 이용하여 표현하며, 이 규칙은 방향 그래프로 저장된다. 바람직한 오프셋 규칙은 별도의 방향 그래프로 저장된다. 모양 규칙도 별도로 저장된다. 이러한 방법은 방향 규칙들과 텍스트 규칙들 모두를 통합하고, 또한 다른 모양 규칙들이 포함되도록 변경될 수 있는 단방향 방법이나 이차 프로그래밍 방법들에 비해 이점들을 갖는다.

또한, 이러한 실시예에서, 각 콘테이너는 박스와 연관된다. 일부 콘테이너의 경우, 추가적인 별도의 박스가 콘테이너와 연결되어 있을 수 있으며, 콘테이너의 이상적인 크기 및/또는 모양을 표현하는데 사용된다. 레이아웃 엔진(105)이 이상적인 모양을 나타내는 박스의 크기 및 모양에 최대한 근접하도록 콘테이너의 프런트 또는 디스플레이된 모양을 나타내는 박스를 생성하도록 하기 위해 최소 오프셋 규칙을 사용한다. 콘테이너의 이상적인 크기와 모양은 연결된 박스에 모양 규칙을 적용함으로써 정의된다. 콘테이너의 크기와 모양이 이상적인 크기와 모양에 최대한 근접하도록 레이아웃 계산방법을 사용한다. 이상적인 크기와 모양이 레이아웃의 이용 가능한 공간에 맞기 않을 수도 있기 때문에 콘테이너를 정확하게 이상적인 크기와 모양으로 만드는 것이 항상 가능한 것은 아니다. 예를 들어, 제한들로 인해 콘테이너가 이상적인 크기 또는 모양이 되지 않을 수 있다.

최소 오프셋 규칙들은 마크들의 가능한 위치들을 한정하는 제한들을 정의하는데 사용되고, 바람직한 오프셋 규칙들은 레이아웃 계산 방법에 의해 최소화된 목적 함수를 정의하는데 사용되고, 모양 규칙들은 계산 방법을 동적으로 가이드하는데 사용된다. 목적 함수를 정의하기 위하여, 각 바람직한 오프셋 규칙이 에너지 값과 연관된다. 바람직한 오프셋 규칙의 에너지는 바람직한 오프셋과 실제 오프셋 간의 차이의 제곱을 반으로 나눈 것이다. 총 에너지라고 불리는 목적 함수는 모든 바람직한 오프셋 규칙의 에너지 합이다. 바람직한 오프셋 규칙에 강도가 지정된 경우, 에너지에 강도를 곱한다. 바람직한 오프셋 규칙은 자연 길이를 가지며 압축하거나 늘리면 팽팽하게 되는 스프링으로 생각할 수도 있다. 바람직한 오프셋 규칙의 장력은 규칙의 강도에 실제 오프셋과 바람직한 오프셋 간의 차이를 곱한 것이다. 규칙의 강도는 스프링의 뻣뻣함에 비유할 수 있다. 레이아웃 엔진(105)은 모든 바람직한 오프셋 규칙의 장력의 균형을 맞춰서 총 에너지를 최소화 하는 과정을 통해서 레이아웃을 계산된다.

도 28은 레이아웃을 계산하는데 포함된 주요 단계들을 도시한다. 레이아웃 계산 방법(2800)은 총 에너지가 최소화되기 전까지 최소 오프셋 규칙중의 어떠한 것도 위반하지 않고 마크를 이동시킨다(즉 마크의 위치를 이동함). 방법(2800)은 단계(2801)에서 시작하는데, 이 단계에서 레이아웃 엔진(105)은 기본 모델 규칙을 위반하지 않고 목적 함수 값을 줄이기 위해 이동할 수 있는 여러 개의 마크들의 집합을 찾는다. 단계(2803)에서, 이러한 찾기가 성공했는지 여부를 판단하기 위해 테스트가 실시되고, 만약 성공한 경우 단계(2805)로 진행하며 실패한 경우 레이아웃 계산이 완료되고 방법(2800)이 중단된다. 단계(2805)에서 총 에너지를 줄이기 위해 적당한 방향으로 적당한 거리만큼 이동한다. 단계(2805) 후에,방법(2800)은 단계(2801)로 반환되는데, 이 단계에서 방법(2800)은 다른 이동 대상 그룹을 찾는다. 이동 대상 마크 그룹의 선정 방법과 마크 이동 거리 계산방법은 아래에 보다 자세히 기술되어 있다. 도 28에 기술된 과정이 종료되었음을 확실히 하기 위해서, 총 에너지에 약간의 변화를 가져오는 마크의 위치의 작은 변화 및/또는 마크의 위치 변화는 무시된다. 일 실시예에서, 위치의 변화들은 전체 논리 단위 수로 제한될 수 있다.

도 29는, 수평 마크의 임의의 그룹들이 이동되기 전에(단계 2903) 수직 마크들이 먼저 이동되는(단계) 것을 제외하고는, 도 28에 도시된 방법과 동일한 절차(2900)를 도시한다. 단계(2905)의 테스트는 단계(2803)의 테스트와 동일한 목적이다. 본 개시의 범위를 벗어나지 않고, 총 에너지를 줄이는 방법을 조합하는 여러가지 가능한 방법들이 존재한다.

도 30a는, 단계(2901-A)와 같은 일 구현예에서 단계(2901)가 어떻게 수행될 수 있는지를 보다 상세하게 도시한다. 단계(3001)에서, 수직 마크들에 대한 푸쉬 그래프가 계산된다. 이 그래프는 어떠한 마크가 지정된 마크에 의해 푸쉬될 수 있는지를 파악하기 위해 사용된다. 일 실시예에서, 이 그래프는 규칙을 표현하는 그래프와 별도의 데이터 구조를 가질 수도 있다. 다른 실시예에서는, 이 두 개의 데이터 구조가 결합될 수도 있다. 단계(3003)에서, 수직 마크와 수평 최소 오프셋 규칙을 표시하는 그래프에서 제1 점을 추출하여 첫번째 수직 마크를 얻는다. 이러한 마크를 얻기 위해 가능한 여러 가지 가능한 순서가 존재한다. 일 실시예에서, 수평 최소 오프셋 규칙을 나타내는 그래프에 마크에 대응하는 점을 저장하는 순서로 마크가 검색된다. 단계(3003)의 다음 단계(3005)로 방법이 진행되는데, 이 단계에서는 단계(3003)에서 선택한 마크에 의해 푸쉬할 마크가 계산된다. 단계(3005)에서는 최소 오프셋 규칙을 위반하지 않고 0 이상의 거리만큼 우측으로 모두 이동할 수 있는 마크 그룹을 결정한다. 이 그룹은 단계(3003)에서 선택된 마크 외에 선택된 마크에 의해 푸쉬될 수 있는 모든 마크 (즉 푸쉬 그래프 상에서 선택한 마크에 대응하는 점에서 도달할 수 있는 점에 대응하는 마크들)로 구성된다. 단계(3007)에서는, 단계(3005)에서 결정된 마크 그룹이 최소 오프셋 규칙을 위반하지 않고 우측으로 이동할 수 있는 최대거리를 결정한다. 단계(3007)의 계산방법은 푸쉬 동작의 단계(2507)의 일부로서 수행되는 방법과 동일하다. 단계(3007)의 다음 단계(3009)에서는 총 에너지를 줄이기 위해 그룹 내 마크들이 이동해야 하는 거리가 계산되는데, 이 거리는 단계(3007)에서 계산된 거리보다 크지 않다. 단계(3009)가 이하에서 보다 상세히 설명되어 있다. 이동된 거리는 정수의 논리 단위로 한정될 수 있다. 단계(3009) 후에, 방법은 단계(3011)로 진행되는데 단계(3009)에서 계산된 거리만큼 마크가 이동된다. 만약 단계(3009)에서 계산한 거리가 단계(3007)에서 계산된 최대허용거리와 같을 경우 푸쉬 그래프가 변경되어야 한다. 단계(3011)의 다음 단계(3013)에서 이 작업이 실시된다. 마크를 우측으로 이동한다고 해서 총 에너지가 감소하지 않을 수도 있다. 이 경우 단계(3009)에서 계산된 거리가 0이 되며 단계(3011)에서 마크가 실제로 위치를 바꾸지 않을 수 있다. 그 결과 단계(3011 및 3013)에서 사실상 하는 것이 없게 되기 때문에 일부 경우 그냥 넘어간다. 이동 방향이 역으로 바뀌었다는 것을 제외하고 단계들(3015, 3017, 3019, 3021 및 3023)은 각각 단계들(3005, 3007, 3009, 3011 및 3013)과 동일하다. 이 단계들에서 푸쉬 그래프 대신에 역 푸쉬 그래프가 사용된다. 소프트웨어 엔지니어링에 익숙한 사람들은 별도의 역 푸쉬 그래프가 필요하지 않도록 동일한 푸쉬 그래프가 역 방향으로 변경되게 푸쉬 그래프의 방향 그래프 데이터 구조를 설계하는 것이 가능함을 알 것이다. 단계(3023) 후 단계(3003)에서, 모든 점을 확인했는지 테스트를 실시하는데, 만약 그렇지 않을 경우 단계(3003)로 돌아가서 다음 마크를 선정한다. 만약 그렇다면, 단계(2901)가 완료되며 계산이 단계(2903)로 진행한다.

단계(2903)는 수직 마크들 및 수평 최소 오프셋 규칙들 대신에 수평 마크들 및 수직 최소 오프셋 규칙들이 고려된다는 점을 제외하고 단계(2901)와 동일하다.

도 31은 단계(3009)의 계산이 어떻게 수행되는지를 상세하게 도시한다. 단계(3101)에서, 현재 위치에서 현재 마크 세트의 위치 변화에 관한 총 에너지의 편도합수가 계산된다. 총 에너지의 편도함수는 장력의 합과 같다. 총 에너지의 편도함수는 현재 세트의 마크에서 현재 세트에 포함되지 않는 마크를 연결하는 바람직한 오프셋 규칙의 합과 같다. 즉 preferred(*n*,*m*,d,s) 형식의 규칙(*m*은 현재 세트 내 마크, *n*은 현재 세트에 없는 마크)에 대한 합니다. 임의의 *m* 및 *n*에 대하여 규칙 preferred(*m*,n,d,s)이 preferred(*n*,*m*,-d,s)와 동일함에 주목할 필요가 있다. 장력의 합을 가능한한 0에 가깝게 함으로써 에너지를 최소화된다. 이는 다음의 공식에 의해 주어지는 거리 δ만큼 마크들을 이동함으로써 단계(3103)에서 수행된다:

$$\delta = -D/S$$

여기서 D는 장력의 합이며, S는 현재 세트 내의 마크를 현재 세트에 존재하지 않는 마크와 연결하는 바람직한 오프셋 규칙의 강도의 합이다. 단계(3105)에서, 고려하고 있는 현재 마크 세트의 이동 방향으로 목적 함수가 감소하고 있는지 여부를 판악하기 위해 테스트를 실시하는데, 이는 δ가 양수인지 테스트 하는 것과 같다. 만약 현재 방향으로 마크를 이동했을 때 에너지가 감소하지 않을 경우, 단계(3017)에서 이동 거리가 0으로 설정된다. 또 현재 방향으로 마크를 이동했을 때 에너지를 감소시킬 수 있을 경우, 단계(3109)에서 이동할 거리가 계산된다. 이동할 거리는 δ보다 작으며 단계(3007)에서 값이 계산된다. 마크들의 이동은 정수 논리 단위로만 제한될 수 있기 때문에, 단계(3007)에서 계산된 거리는 정수 논리 단위로 반올림된다. 이는 레이아웃 계산의 종료를 보장한다.

원본 마크들에 대한 동일한 고려사항들이 앞서 기재된 푸쉬 작동에 적용된는 것처럼 레이아웃 계산에 적용된다. 이러한 점에서, 레이아웃 계산 시 원본 마크를 무시할 수 있는데, 이 경우 단계(3005)에서 수평 좌표 시스템의 원 시작을 정의하는 원 수직 마크가 이동할 마크 세트에 포함될 수도 있는 경우를 제외하고 항상 이동할 수 있는 마크 세트를 찾게 된다. 이 때 단계(3007)에서는 항상 0 외의 거리가 나오게 되며, 단계(3011)에서 원래 마크의 위치가 변할 수 있다. 이 경우 단계(3001) 전에 원 거리의 위치가 저장되어야 하며 계산동안 원래 마크가 이동한 거리를 빼서 레이아웃 상의 모든 수직 마크를 이동함으로써 원래 마크를 복원해야 한다. 단계(3007)에서 원본 마크를 고려하는 경우도 있다. 이에 관해, 이동될 마크들의 세트에 포함되면, 단계(3007)에서 계산된 거리는 0이다. 이 경우, 원본은 레이아웃 계산을 수행할 때 저장되거나 복원될 필요가 없다.

모양 규칙들은 비고정된 폭 및 비고정된 높이 모두를 갖는 콘테이너들에 사용될 것을 필요로 한다. 이러한 콘테이너의 경우, 콘텐츠들을 담고 있는 콘테이너 부분이 두 개 박스로 레이아웃 엔진(105)에 의해 표현된다. 한 개 박스는 콘테이너 내 포함된 디스플레이 되거나 프린트된 텍스트 또는 이미지의 경계를 나타낸다. 이 박스의 경계는 레이아웃 엔진에 의해 계산된다. 다른 나머지 박스는 이상적인 박스 모양이다. 이 박스는 모양 규칙에 의해 제어되며, 일반적으로 사용자에게 프린트되거나 디스플레이되지 않는다. 두 개의 박스는 한개의 수평 마크와 한개의 수직 마크를 공유한다. 예를 들어, 두 개 박스의 좌측 상부 에지의 위치가 항상 동일하다. 좌측 또는 우측 에지를 공유한 경우 또는 상부 또는 하부 에지를 공유하는 경우에 차이가 없다. 나머지 두개의 수직 에지는 최소 거리 규칙과 관련되며, 마찬가지로 나머지 두개의 수평 에지는 최소 거리 규칙과 관련된다. 최소 거리 규칙들은 두개의 박스들이 가능한한 크기가 서로에 대해 근접해야함을 레이아웃 엔진에 지시한다.

관련된 한쌍의 박스들 중 한 박스의 에지와 연관된 임의의 마크를 이동시키는 경우, 모양 규칙들을 고려한다. 공유되지 않은 모양 규칙을 가진 박스에 속한 마크를 단계(3003)에서 무시하고 그냥 넘어갈 수 있지만, 모양 규칙을 갖지 않은 관련된

박스의 에지를 나타내는 임의의 마크들을 이동시키려면 단계(3009) 이전에 이처럼 공유되지 않은 마크들이 조정된다. 수평 마크에 적용되는 동일한 단계들에 이 사항이 동일하게 적용된다. 에너지 함수를 최소화하기 위해 모양 규칙에 맞게 공유하지 않는 마크를 조정한다. 이러한 처리는 에너지가 수직 마크와 수평 마크 간에 밸런싱됨을 보장한다.

모양 규칙이 방향 규칙인 경우, 에너지를 최소화하는 방법이 명확하다. 만약 디스플레이 박스의 폭과 높이가 각각 W와 H이고 이상적인 박스의 폭과 높이가 w와 h인 경우 W+H=w+h일 때 최소 에너지가 달성된다. 만약 면 규칙에 의해 정의된 폭 대비 높이의 비율이 r이면 이상적인 박스의 폭과 높이는 다음과 같이 계산될 수 있다:

$w = (1+r)^{-1}(W+H)$ 및 $h = r(1+r)^{-1}(W+H)$.

모양 규칙이 텍스트 규칙인 경우, 이상적 박스의 모양에 대한 어떠한 간단한 공식도 존재하지 않는다. 텍스트 레이아웃 엔진은 직사각형 공간 내에 있는 레이아웃 텍스트를 계산한다. 텍스트 레이아웃 엔진은 이상적인 박스의 차원을 계산하는데 사용된다. 주어진 폭을 갖는 공간에 텍스트 블록을 배치하기 위해 텍스트 레이아웃 엔진이 계산을 수행하며, 이 엔진은 레이아웃 텍스트 블록의 높이를 신속하게 계산할 수 있다. 텍스트 레이아웃 엔진은 장문의 텍스트를 여러 라인에 랩하기 위해 워드-랩 계산을 포함한다. 텍스트 레이아웃 엔진은 텍스트와 관련된 포맷팅 정보에 맞게 텍스트를 레이아웃 한다. 그 포맷팅 정보는 예를 들어, 각각의 문자에 대해 사용하기 위한 볼드체 또는 밑줄과 같은 스타일, 폰트의 크기 및 단란 스타일를 포함할 수 있다.

일 실시예에서, 이상적인 폭은 가장 넓은 텍스트 라인의 폭으로 세팅되고, 이상적인 높이는 텍스트 레이아웃 엔진에 의해 계산된 레이아웃된 텍스트 블록의 높이로 설정되는데, 이 경우 높이는 디스플레이 박스의 높이와 동일하다. 이는 폭이 가장 긴 라인 폭보다 적거나 또는 디스플레이 박스가 텍스트를 포함하기에 충분히 높지 않다면, 에너지 페널티를 추가하는 효과를 갖는다.

또 다른 실시예에서, 텍스트 레이아웃 엔진은 가장 낮은 에너지 값을 초래하는 폭이 찾아질 때까지 각 폭에 대한 높이를 계산하는 다른 크기의 폭들로 그 텍스트를 레이아웃하는데 사용된다. 이 때 텍스트의 가장 넓은 라인이 디스플레이 박스만큼 넓지 않을 경우 이상적인 높이는 텍스트 라인 중 가장 넓은 것에 맞게 설정되고 라인의 랩이 없을 경우 이상적인 높이는 레이아웃 엔진에 의해 계산된 높이로 세팅 된다. 그러나 텍스트가 디스플레이 박스에 포함되기 위해서 랩을 해야 하는 긴 라인을 포함하는 경우, 최소한의 에너지를 요구하는 레이아웃 폭을 찾기 위한 전략이 사용된다. 본 개시의 범위를 벗어나지 않고도 이상적인 폭을 찾기 위한 많은 가능한 전략들을 고안할 수 있다. 특히, 바이너리 검색 전략이 사용될 수 있다.

또 다른 실시예에서, 디스플레이 박스에 맞도록 랩핑되어야 하는 텍스트의 긴 라인들을 텍스트가 포함한다면, 그 이상적인 박스의 차원들은 텍스트가 디스플레이 박스에 맞도록 레이아웃될 수 있는지의 여부에 기초하여 계산된다. 만약 텍스트가 디스플레이 박스에 들어갈 수 없으면, 폭과 높이 모두에 있어서 디스플레이 박스보다 넓은 이상적인 박스가 만들어진다. 텍스트가 디스플레이 박스 내에 들어간다면, 폭 및 높이에 있어 디스플레이 박스보다 작은 이상적인 박스가 만들어 진다. 이러한 전략은, 디스플레이 박스가 텍스트를 포함하기 위한 필요보다 작거나 클 때 레이아웃에 페널티를 추가하는 것과 같은 효과를 낸다. 본 개시의 범위를 벗어나지 않고 이상적인 박스의 정확한 차원을 계산하는 많은 가능한 방법이 존재한다. 사용 가능한 하나의 특정한 방법은, 디스플레이 박스의 현재 폭과 동일한 폭으로 레이아웃된 텍스트의 높이를 우선 계산하고, 이상적인 박스의 차원을 구하기 위해 레이아웃된 박스의 높이와 디스플레이 박스의 높이의 차이의 절반을 디스플레이 박스의 폭 및 높이 모두에 더하는 것이다.

18.3.1 단순 1차원 레이아웃 예시

도 39는 섹션 18.3 및 다음 항에 기재된 레이아웃 메카니즘의 예시적인 사용을 도시한다. 세개의 박스들(3901)은 A, B 및 C로 지정된다. 각 박스(3901)는 좌측, 상부, 우측, 하부 에지들에 마크들(3905)로 정의된다. 페이지의 좌측 및 우측 에지들은 그 마크들 간의 고정된 거리로 두개의 수직 마크들(3903 및 3904)로 정의된다. 일부 콘테이너의 에지들과 페이지의 에지 간의 거리들을 지정하는 고정된 오프셋 규칙들(3902)이 존재한다.

상세한 규칙들은 다음과 같다:

- 페이지의 폭은 50 단위로 고정된다.
- 콘테이너 A의 바람직한 폭, P_A는 22 단위이다.
- 콘테이너 B의 바람직한 폭, P_B는 16 단위이다.
- 콘테이너 C의 바람직한 폭, P_C는 13 단위이다.
- 박스 A의 좌측 에지(도 39에서 'a'로 언급됨)는 페이지 좌측에서 1 단위에 위치한다.
- 박스 A의 우측 에지('m'으로 언급됨)는 이동이 자유롭다.
- 박스 B의 좌측 에지('b'으로 언급됨)은 'a'의 6 단위 우측에 있다.
- 박스 B의 우측 에지는 'm'의 1 단위 좌측에 있다.
- 박스 C의 좌측 에지는 'm'의 2 단위 우측에 있다.

● 박스 C의 우측 에지('c'로 언급됨)는 페이지 우측 에지의 1 단위 좌측에 있다.

박스들 A, B, C의 현재 폭들이 각각 W_A, W_B, W_C로 표현되도록 한다. 편의를 위해 좌표 시스템을 사용하여 페이지 좌측 에지를 0으로 하는 것에서 출발해서 페이지 우측으로 이동할수록 증가하도록 한다. 또한 문자들 'a', 'b', 'c', 'm'은 좌표 시스템 내의 대응하는 마크들의 위치를 의미하는 것으로 이해될 수 있는 것으로 가정한다.

바람직한 오프셋 규칙의 에너지는 바람직한 오프셋과 실제 오프셋 간의 차이의 제곱의 반이다. 총 에너지인 이 레이아웃의 목적 함수는 모든 바람직한 오프셋 규칙의 에너지의 합이다. 그래서 목적 함수는 다음과 같다:

$E(A,B,C)=\frac{1}{2}(W_A-P_A)^2+\frac{1}{2}(W_B-P_B)^2+\frac{1}{2}(W_C-P_C)^2$

$=\frac{1}{2}(W_A-22)^2+\frac{1}{2}(W_B-16)^2+\frac{1}{2}(W_C-13)^2$

$=\frac{1}{2}[(W_A-22)^2+(W_B-16)^2+(W_C-13)^2]$

$=\frac{1}{2}[((m-a)-22)^2+((m-b-1)-16)^2+((c-m-2)-13)^2]$

$=\frac{1}{2}[(m-a-22)^2+(m-b-17)^2+(c-m-15)^2]$.

각 박스의 폭을 찾기 위해, 그 박스의 가장 좌측 마크 위치가 음이 아닌 수를 얻기 위해 가장 우측 마크의 위치로부터 감산되었음을 유의해야 한다. 그러므로, W_B는 (m-b-1)인데 그 이유는 'm'과 박스 B의 우측 에지 간의 갭이 1 단위이기 때문이다. 그리고 W_C는 (c-m-2)인데 왜냐하면 'c'가 박스 C의 좌측 에지와 마크 'm' 간에 거리가 2 단위이기 때문이다.

목적 함수를 최소화 하기 위해서 이동할 수 있는 마크는 'm' (실제로 'm'과 그에 가장 가까운 두 개 마크가 이동할 수 있으며 이들 모두 공식의 통제 변수로 취급할 수 있기 때문에 사항을 단순화 하기 위해 'm'으로 언급된 것이 사용된다) 임을 유의해야 한다. 이 경우 변수 m과 관련된 에너지 함수의 편도함수만 찾으면 충분하며, 편도함수가 0인 경우에 대해 풀면 된다.

$E(A,B,C)=\frac{1}{2}[(m-a-22)^2+(m-b-17)^2+(c-m-15)^2]$

$\delta E/\delta m = (m-a-22)+(m-b-17)+(m-c+15)$.

여기서 셋째 항의 부호 (m-c+ 15)가 바뀜에 주목할 필요가 있는데 이 항에 음수인 m과 관련하여 편도함수가 결정됨을 주목할 필요가 있다.

$\delta E/\delta m = m - a - 22 + m - b - 17 + m - c + 15$

$= 3m - a - b - c - 22 - 17 + 15$

$= 3m - a - b - c - 24$.

이 값을 0으로 해서 풀면 최소 에너지를 찾는 것이 이제 가능해진다. a, b, c가 각각 1, 7, 49의 값을 갖는다는 것이 알려져 있다.

$0 = 3m - 1 - 7 - 49 - 24$

$3m = 1 + 7 + 49 + 24$

$= 81$

$m = 27$.

단지 하나의 변수가 존재하기 때문에, 이는 그 절차를 종료한다. 그러므로, 주어진 구성의 가장 낮은 에너지는 마크 'm'이 그 페이지의 좌측 에지의 27 단위 우측에 있을 경우이다.

18.3.2 간단한 이차원 레이아웃 예시

도 40a는 레이아웃 메카니즘 사용의 또 다른 예를 도시한다. 이 경우, 움직일 수 있는 수평 마크와 수직 마크 둘 모두 존재하기 때문에, 그 예는 이차원의 장력들을 포함한다.

마크(4002)로 경계된 알려져 있는 고정 크기의 직사각형 페이지 내에 도시된 A, B 및 C로 라벨링된 세개의 박스들(4001)이 존재한다. 고정 오프셋 규칙들(4003, 4004, 4005 및 4006)은 A와 C의 세개의 에지들의 위치를 결정한다. A의 하부 수평 에지가 이동할 수 있지만 고정 오프셋 규칙에 의해 B의 상부 에지 ('m'으로 언급됨)에 연결되어 있기 때문에 이 두 개의 마크는 함께 움직이고 동일한 거리로만 움직일 수 있다. 마찬가지로, 그들 간에 고정된 오프셋 규칙이 있기 때문에, B의 우측 에지('n'으로 언급됨)와 C의 좌측 에지가 이동될 수 있지만 동일한 거리만큼 함께 움직일 수 밖에 없다.

박스 B의 폭과 높이가 변할 수 있기 때문에, 모양 규칙이 박스 B에 적용된다. 여기서 선택한 규칙은 면 비율 규칙으로 이 규칙은 박스 B의 폭 대비 높이의 비율이 가능하면 0.5가 되도록 규정한다(이는 도 40a에서 $R_B = 0.5$으로 라벨링된 화살표로 표시된다). 박스 A는 22 단위의 바람직한 높이를 가지며, 박스 C는 36 단위의 바람직한 폭을 갖는다. (규칙들 4005 및 4006로 인해)박스 A 및 박스 C 어느 것도 이차원으로 변경할 수 없으므로, 모양 규칙을 갖지 않는다.

상세한 규칙들은 다음과 같다:

● 페이지 폭이 50 단위로 고정된다.

● 페이지의 높이는 36 단위로 고정된다.

● 콘테이너 A의 바람직한 높이, P_A는 22 단위이다.

● 콘테이너 B의 높이 대 폭의 바람직한 비율, R_B는 0.5이다.

● 콘테이너 C의 바람직한 폭, P_C는 36 단위이다.

● 박스 A의 상부 에지('a'로 언급됨)는 페이지 상부에서 2 단위에 있다.

● 박스 A의 좌측 에지는 페이지의 좌측 에지의 1 단위 우측에 있다.

● 박스 A의 우측 에지는 좌측 에지의 18 단위 우측에 있다.

● 박스 B의 상부 에지('m'으로 언급됨)는 이동할 수 있다.

● 박스 A의 하부 에지는 'm'보다 2 단위 위에 있다.

● 박스 B의 좌측 에지('b'로 언급됨)는 페이지의 좌측 에지의 1 단위 우측에 있다.

● 박스 B의 하부 에지('d'로 언급됨)는 페이지 하부 에지보다 3 단위 위에 있다.

● 박스 B의 우측 에지('n'으로 언급됨)는 움직일 수 있다.

● 박스 C의 좌측 에지는 'n'의 2 단위 우측에 있다.

● 박스 C의 하부 에지는 페이지 하부 에지보다 3 단위 위에 있다.

● 박스 C의 우측 에지('c'로 언급됨)는 페이지 우측 에지의 2 단위 좌측에 있다.

● 박스 C의 상부 에지는 하부 에지의 16 단위 위에 있다.

H_A, W_A, H_B, W_B, H_C, W_C가 각각 박스 A의 높이 및 폭, 박스 B의 높이 및 폭, 박스 C의 높이 및 폭이라고 하자.

P_{HB} 및 P_{WB}가 각각 박스 B의 바람직한 높이 및 폭이라고 하자. 박스 B의 바람직한 높이들 및 폭들은 레이아웃 알고리즘에 고정되지 않는다. 대신에 그들은 박스 B의 모양 규칙의 사용에 의해 임의의 단계들에서 계산되며, 이는 폭 및 높이에 관한 종횡비 규칙이다. 이 값들은 박스 B의 높이와 폭의 가중된 평균들로 계산된다:

$$P_{WB} = (W_B + W_B) / (1 + R_B) = \tfrac{2}{3}(W_B + H_B)$$

$$P_{HB} = R_B(W_B + H_B) / (1 + R_B) = \tfrac{1}{3}(W_B + H_B).$$

이러한 배열의 에너지를 표현하는 목적 함수는 수평 방향 및 수직 방향 둘 모두에서 장력 제곱의 합의 반으로 얻어질 수 있다:

$E(A,B,C) = \frac{1}{2}(H_A-P_A)^2 + \frac{1}{2}(H_B-P_{HB})^2 + \frac{1}{2}(W_B-P_{WB})^2 + \frac{1}{2}(W_C-P_C)^2$

$= \frac{1}{2}[(H_A-P_A)^2 + (H_B-P_{HB})^2 + (W_B-P_{WB})^2 + (W_C-P_C)^2]$

$= \frac{1}{2}[(m-2-a-P_A)^2 + (d-m-P_{HB})^2 + (n-b-P_{WB})^2 + (c-n-2-P_C)^2]$.

(에너지를 실제로 계산할 필요없이)이 에너지를 감소시키기 위해서는, 수직적 차원과 수평적 차원 모두에서, 장력의 절대값의 합이 더 감소될 수 없을 때까지, 차례로 목적 함수의 편미분값들을 감소시키는 것으로 충분하다. 이를 통해 가장 장력이 작은 상황을 얻을 수 있다. 음 및 양의 장력들이 중단 조건에 있어서 동일하게 처리될 수 있도록, 장력의 절대값을 사용한다. 편미분들은 다음과 같다:

$\delta E/\delta m = m - 2 - a - P_A + m - d + P_{HB} = 2m - 59 + P_{HB}$

$\delta E/\delta n = n - c + 2 + P_C + n - b - P_{WB} = 2n - 11 - P_{WB}$

이 편미분값들을 각각 0으로 설정하여 그 차이들을 최소화하면, 다음의 관계들이 얻어진다:

$m = \frac{1}{2}(2 + a + P_A + d - P_{HB}) = \frac{1}{2}(59 - P_{HB})$

$n = \frac{1}{2}(c - 2 - P_C + b + P_{WB}) = \frac{1}{2}(11 + P_{WB})$

예시를 위하여, 박스 B의 초기 폭 및 높이가 각각 15와 10 단위로 하면, $m = d - 10 \equiv 33 - 10 = 23$이고 $n = b + 15 = 16$이다.

$W_B = 15$

$H_B = 10$

$m = 23$

$n = 16$.

박스 B의 바람직한 폭 및 높이가 계산될 수 있다:

$P_{WB} = \frac{2}{3}(W_B + H_B)$

$= \frac{2}{3}(15 + 10)$

$= 50/3$

$= 16\frac{2}{3}$

= 16(가장 근접한 정수 단위로 감소되는 경우).

$P_{HB} = \frac{1}{3}(W_B + H_B)$

$= \frac{1}{3}(15 + 10)$

$= 25/3$

$= 8\frac{1}{3}$

= 8(가장 근접한 정수 단위로 감소되는 경우).

레이아웃 메카니즘을 언제 종료할지를 결정할 수 있도록, 수직 및 수평 장력의 절대값의 합이 계산된다. 이 경우 수평 장력 T_X는 박스 B와 박스 C의 현재 폭과 바람직한 폭 간의 차이의 합이다. 유사하게 수직 장력 T_Y는 박스 A와 박스 B의 현재 높이와 바람직한 높이 간의 차이 합이다.

$T_X = (W_C - P_C) + (W_B - P_{WB})$

$= (c - n - 2 - 36) + (n - b - P_{WB})$

$= c - 38 - b - P_{WB}$(n이 제거된 경우)

$= 48 - 38 - 1 - P_{WB}$

$= 9 - P_{WB}$

$= 9 - 16$

$= -7.$

$T_Y = (H_B - P_{HB}) + (H_A - P_A)$

$= (d - m - P_{HB}) + (m - 2 - a - 22)$

$= d - P_{HB} - 2 - a - 22$(m이 제거된 경우)

$= 33 - P_{HB} - 2 - 2 - 22$

$= 7 - P_{HB}$

$= 7 - 8$

$= -1.$

장력이 음이라는 사실은 이러한 장력들을 감소시키기 위해 마크들이 좌측 또는 위로 이동할 필요가 있을 수 있음을 나타낸다.

총 장력은 수직 및 수평 장력의 절대값들의 합이다:

$T_{TOTAL} = |T_X| + |T_Y|$

$= |-7| + |-1|$

$= 7 + 1$

$= 8.$

이러한 총 장력이 더 이상 감소될 수 없을 때까지, 차례로 수평 및 수직 차원들로 장력들을 감소시키도록 레이아웃 메카니즘이 진행된다.

먼저 수직 마크는 수평 방향으로 이동한다. B의 바람직한 폭은 이전에 다음과 같이 계산되었다.

$P_{WB} = \frac{2}{3}(W_B + H_B)$

$= 16.$

움직일 수 있는 수직 마크들만이 'n'이고, 그 마크는 고정된 오프셋 규칙에 의해 (우측으로 2 단위) 그에 접속된다. 두개의 마크들은 수평 장력을 감소시키는 방향으로 이동된다. 다른 마크의 위치가 (n+ 2)로부터 추론될 수 있기 때문에 'n'이 이동되어야 하는 위치를 결정하기에 충분하다. 다음의 관계식에 의해 이전에 결정된 바와 같이, n에 대한 목적 함수의 편도함수를 감소시키기 위하여 'n'의 위치를 계산한다.

$n = \frac{1}{2}(11 + P_{WB})$

$= \frac{1}{2}(11 + 16)$

= 13½

= 13(가장 근접한 정수 단위로 감소되는 경우).

수평 장력을 줄이기 위해 마크 'n'이 위치 13으로 이동되며, 이는 또한 박스 B의 폭을 12로 변경시킨다:

$W_B = 12$

$H_B = 10$

m = 23

n = 13.

박스 B의 바람직한 폭 및 높이가 모양 규칙에 기인하여 변경된다:

$P_{WB} = ⅔(W_B + H_B)$

= ⅔(12 + 10)

= 44/3

= 14⅔

= 14(가장 근접한 정수 단위로 감소되면)

$P_{HB} = ⅓(W_B + H_B)$

= ⅓(12 + 10)

= 7⅓

= 71(가장 근접한 정수 단위로 감소되면)

총 장력이 감소되었는지 확인하기 위해서 총장력을 다시 계산해야 하며, 아니라면, 레이아웃 절차는 중지할 것이다. 새로운 장력은 다음과 같다.

$T_X = 9 - P_{WB}$

= 9 - 14

= -5.

$T_Y = 7 - P_{HB}$

= 7 - 7

= 0.

$T_{TOTAL} = |T_X| + |T_Y|$

= |-5| + |0|

= 5.

총 장력이 8에서 5로 감소되었기 때문에, 그 절차가 계속될 것이다.

이제, 수평 마크들이 수직 방향으로 이동된다. 이 경우에서, 'm'과 그에 접속된 마크가 이동할 수 있다. 그 접속된 마크가 (m-2)로 이동할 것이기 때문에, 'm'이 이동되어야 하는 위치를 계산하기에 충분하다. 다음의 관계식에 의해 앞서 결정된 바와 같이, 목적 함수의 편도함수는 m에 대해 최소화된다.

$m = \frac{1}{2}(59 - P_{HB})$

$= \frac{1}{2}(59 - 7)$

$= 26.$

수직 장력을 감소시키기 위해 마크 'm'은 위치(26)로 이동되며, 고정된-오프셋 규칙에 의해 그에 접속된 마크는 위치(24)로 이동한다.

$W_B = 12$

$H_B = 7$

$m = 26$

$n = 13$

박스 B의 바람직한 폭, 높이 및 총 장력은 다음과 같이 재계산된다:

$P_{WB} = \frac{2}{3}(W_B + H_B)$

$= \frac{2}{3}(12 + 7)$

$= 38/3$

$= 12\frac{2}{3}$

$= 12$(가장 근접한 정수 단위로 감소되는 경우).

$P_{HB} = \frac{1}{3}(W_B + H_B)$

$= \frac{1}{3}(12 + 7)$

$= 6\frac{1}{3}$

$= 6$(가장 근접한 정수 단위로 감소되는 경우).

$T_X = 9 - P_{WB}$

$= 9 - 12$

$= -3.$

$T_Y = 7 - P_{HB}$

$= 7 - 6$

$= 1.$

$T_{TOTAL} = |T_X| + |T_Y|$

$= |-3| + |1|$

$= 4.$

총 장력이 5에서 4로 줄었기 때문에, 그 절차는 계속될 것이다. 다시 수직 마크들은 장력을 감소시키기 위해 수직 마크들이 수평 방향으로 이동된다.

$n = \frac{1}{2}(11 + P_{WB})$

= ½(11 + 12)

= 11½

= 11(가장 근접한 정수 단위로 감소되는 경우).

마크 'n'은 위치11로 이동된다. 박스 B의 폭은 10 단위가 된다.

$W_B = 10$

$H_B = 7$

$m = 26$

$n = 11$

박스 B의 바람직한 폭 및 높이와 총 장력이 재계산된다:

$P_{WB} = ⅔(W_B + H_B)$

= ⅔(10 + 7)

= 34/3

= 11⅓

= 11(가장 근접한 정수 단위로 감소되는 경우).

$P_{HB} = ⅓(W_B + H_B)$

= ⅓(10 + 7)

= 5⅔

= 5(가장 근접한 정수 단위로 감소되는 경우).

$T_X = 9 - P_{WB}$

= 9 - 11

= -2.

$T_Y = 7 - P_{HB}$

= 7 - 5

= 2.

$T_{TOTAL} = |T_X| + |T_Y|$

$= |-2| + |2|$

= 4.

총 장력이 4 이하로 감소되지 않았으므로, 절차는 여기서 중지할 수 있다. 대안으로, 절차가 중지하기 전에 수평 마크들이 또한 이동될 수 있다.

이 예제의 최종 결과는 도 40b와 같을 수 있다. 박스 A와 박스 C가 이 경우에 겹칠 수 있음에 유의하라. 이것은 원하는 결과가 될 수 있다. 그렇지 않으면, 이를 방지하기 위해 다른 제한을 추가해야 할 수도 있다. 예를 들어, 박스 C의 좌측 에지가 박스 A의 우측 에지의 우측에 있도록 최소 오프셋 규칙이 추가될 수 있다.

이 예는 하나의 모양 규칙 및 두개의 바람직한 오프셋 규칙만을 보여주지만, 실제 애플리케이션은 박스들 각각에 대해 모양 규칙들을 사용할 수 있다. 모양 규칙들은 또한 박스들 내에 디스플레이되어야 할 콘텐츠들에 의존하여 달라질 수 있다. 예를 들어, 종횡비 규칙들 및 텍스트 모양 규칙들 둘 모두가 사용될 수 있다. 각각은 자신의 방식대로 박스의 이상적인 모양을 계산할 것이다.

18.4 고정 중심들을 갖는 그래프 기반 알고리즘

레이아웃 엔진(105)에서, 고정된 중심 규칙들의 사용이 허용되므로, 고정된 중심 규칙들을 올바르게 처리하기 위해 레이아웃 계산을 변경할 필요가 있다. 단계(2901)의 레이아웃 계산에 대한 변경은 도 30b에서 단계(2901-B)로 도시되고, 이는 푸쉬 동작에서 고정된 중심 규칙을 지원하기 위한 변경과 유사하다. 도 30b는 도 30a와 거의 동일하지만, 레이아웃을 계산할 때 어떻게 고정된 중심이 고려되었는지를 도시한다. 단계들(3001, 3003, 3005, 3009, 3013, 3015, 3019, 3023 및 3025)은 도 30a와 도 30b에 공통이다.

도 30b에서, 푸쉬된 마크들을 계산할 때 단계들(3005 및 3015)은 반대 마크들을 포함해야 한다. 한 방향으로 이동할 단일 마크 세트를 계산하는 대신 두 개의 마크 세트가 계산된다. 즉 우측으로 이동해야 할 한 개 세트와 좌측으로 이동할 한 개 세트이다. 이것은 푸쉬 동작과 마찬가지이다. 도 30b에서는, 단계(3008)가 단계(3007)를 대체하는데, 이 단계에서 (고정된 중심 규칙을 무시한 채로) 두 개의 마크 세트에 독립적으로 이동할 최대 거리가 계산되고, 그 다음 두 값 중 (크기가) 최소값이 마크가 움직일 수 있는 최대 거리가 된다. 단계(3011)는 단계(3012)로 대체되고, 단계(3021)는 단계(3022)로 교체되는데, 차이는 두 개 마크 세트 모두 이동된다는 것이다. 단계(3012)에서는, 단계(3003)에서 선택한 마크를 포함한 세트가 우측으로 이동하고, 첫번째 세트의 반대 마크를 포함한 세트는 우측으로 이동된다. 고정된 중심을 푸쉬 동작에 포함되도록 변경할 때도 동일한 사항이 적용된다. 이런 점에서 만약 두 개의 반대 마크가 동일한 세트에 들어있다면 원래 마크를 두 개 세트에 추가해야 한다. 논리적으로, 원본을 이동시키는 것은 모든 고정 마크들이 이동되어야 하고, 따라서 고정 중심을 갖는 모든 박스들도 또한 이동해야 함을 의미하므로, 원본 마크가 포함되어 있다면, 동일한 방위를 갖는 반대 마크의 모든 쌍들이 두 세트 모두에 포함되어야 한다. 물론, 원본 마크와 반대 마크들의 모든 쌍들을 포함하는 세트를 실질적으로 생성할 필요는 없는데, 이는 이러한 세트들이 이동되어서는 안되기 때문이다. 원본 마크가 둘 중의 한 세트에 포함되어 있다면, 단계(3008)은 최대 거리를 0으로 설정할 것이므로, 단계(3009, 3012 및 3013)은 아무런 작업도 수행하지 않고, 생략될 수 있다. 단계(3018)는 단계(3017)를 대체하며, 단계(3008)와 같이, 최대 거리는 단계(3003)에서 선택된 마크를 포함하는 두 세트 모두에 대해 계산되지만, 또한 선택한 마크를 포함하는 세트 내의 마크들에 대향하는 모든 마크들에 대해서도 계산된다. 원본 마크가 두 세트 중의 하나에 포함되어 있다면, 단계(3018)는 마찬가지로 최대 거리를 0으로 설정해야 한다.

19. 문서 프린트

도 36은 문서 탬플릿으로부터 문서들을 생성하고 프린트할 때 포함되는 방법(3600)의 단계들을 도시한다. 방법(3600)은, 애플리케이션(121)이 데이터베이스에 연결되고, 필요한 소스 데이터로 구성된 표의 시작에 데이터베이스 커서를 설정하는 단계(3601)에서 시작된다. 사용자는 도 14에 도시된 것과 같은 다이알로그를 통해서 연결할 데이터베이스가 어떤 것인지를 지정할 수 있다. 단계(3601)의 다음 단계(3603)에서, 데이터베이스 테이블에서 다음 레코드를 가져오고, 커서가 다음 레코드를 가리킬 수 있도록 업데이트 된다. 문서 세트의 각 문서가 한 개의 레코드에 대응되기 때문에, 새로운 문서가 시작되면 페이지 카운터가 문서 탬플릿의 시작으로 재설정된다. 그 다음 단계(3605)에서는, 애플리케이션(121)이 생성될 문서의 새로운 페이지를 시작되고, 해당 페이지에 대한 레이아웃이 소스되어 페이지에 적용된다. 단계(3607)에서, 현재 페이지의 레이아웃 내의 각 콘테이너의 이상적 크기를 계산하기 위해 현재 레코드의 데이터가 사용된다. 단계(3607)에서 계산된 이상적 크기는, 단계(3609)에서 레이아웃 규칙을 조정하기 위해 사용된다. 이러한 동작은, 바람직한 오프셋 규칙의 값을 변경하고, 단계(3607)에서 계산된 이상적인 크기에 기초하는 종횡 규칙을 업데이트하는 것을 포함한다. 또한, 텍스트의 폭 또는 높이가 고정되지 않은 가변 텍스트 콘테이너의 경우, 단계(3603)에서 가져온 데이터베이스 레코드 필드의 텍스트에 의해 의존하는 텍스트 규칙과 연관하여 폭과 높이가 결정된다. 이러한 텍스트 규칙은 단계(3609)에서도 업데이트 된다. 단계(3609)의 다음 단계(3611)에서, 현재 페이지에 대한 레이아웃이 계산된다. 레이아웃이 계산되면, 단계(3613)에서 페이지가 렌더링 및 프린트된다. 페이지를 프린트 한 후, 단계(3615)에서 테스트를 실시하여 현재 문서의 모든 페이지가 프린트되었는지를 확인한다. 만약 그렇지 않다면, 방법(3600)은 다음 페이지에 대한 계산을 시작하기 위해 단계(3605)로 반환된다. 만약 현재 문서에 대한 문서 탬플릿의 모든 페이지가 프린트 된 경우, 방법(3600)은 단계(3617)로 진행되는데, 이 단계에서 현재 레코드가 데이터베이스 테이블의 마지막 레코드인지를 확인하기 위해 테스트가 실시된다. 테이블의 모든 레코드가 처리되었으면, 방법(3600)이 완료되고, 그렇지 않으면 그 절차는 단계(3603)로 귀환하는데, 이 단계에서 다음 문서가 프린트될 수 있도록 테이블의 다음 레코드를 가져온다.

20. 가능한 텍스트 모양들의 사전 계산

섹션 18에 기재된 그래프-기반의 레이아웃 방법들과 같은 레이아웃 메카니즘의 속도를 높일 수 있는 한가지 방법은 레이아웃에 포함될 임의의 텍스트의 모든 모양들을 미리 계산하는 것이다. 레이아웃 계산 작업 전에 모양을 계산함으로써 레이아웃 메카니즘이 보다 빨리 진행될 수 있는데, 그 이유는 텍스트 모양 계산이 이미 이루어졌고 신속하게 나중에 추출될 수 있도록 결과를 저장했기 때문이다. 일부 텍스트의 가능한 모양들을 계산하는 방법은 도 41a 내지 도 41k에 예로서 도시된다.

그 방법은 다음 방식으로 작용한다:

1. 도 41a와 같이, 그 텍스트의 워드를 수평적으로 끝에서 끝까지 요구되는 워드들 사이에 적당한 공간을 띄어서 배치한다 (각각의 워드(4102)는 경계 박스(4103) 내에 도시됨). 총 워드 구성의 폭과 높이는 워드들을 포함하는 박스 중 가장 작은 박스(4101)로 결정할 수 있다. 경계 박스(4103)의 크기는 메모리 데이터 구조에 저장된다. 필요하면 별개의 폰트를 사용하여 워드를 측정할 수도 있다. 구분을 위해 경계 박스(4103)와 워드 박스들(4102) 간에 작은 공간이 도시되지만, 이 갭은 실제 구현에서는 존재하지 않는 것임을 유의해야 한다.

2. 경계 박스의 폭이 얻어지고, 그 폭은 사용된 좌표 시스템의 1 단위 만큼 감소된다. 그 다음 새로운 폭을 폭 제한으로 상요하여 다시 텍스트를 배치한다. 이 폭 제한과 텍스트 언어의 텍스트 흐름 규칙 때문에 텍스트가 필요하면 다음 라인으로 넘어가게 될 것이다. 높이 제한은 없기 때문에 필요한 만큼 수의 라인을 사용할 수 있다. 도 41b에서와 같이 결과에서 새로운 경계 박스가 정해지고 이 경계 박스도 메모리 데이터 구조에 저장된다. 워드 간의 공간(4105)은 텍스트 언어의 자연적인 공간 띄우기 규칙을 따른다. 예를 들어 영어의 경우 라인 끝의 공간을 경계 박스 계산에 포함하지 않는다. 일부 언어들에 대해 무엇이 워드를 구성하고 무엇이 워드간 공간을 구성하는지의 정의는 주어진 예에따라 달라질 수 있다.

3. 임의의 더 좁은 경계 박스에서 텍스트가 레이아웃될 수 없을 때까지 단계 2가 반복된다. 도 41c 내지 도 41k가 이러한 프로세스를 예시한다. 그 예에서 워드 'Brevity'와 같은 각각의 워드가 내부에 선택적 하이픈 구분점(4104)을 가질 수 있으며, 이는 워드가 도 41h 내지 도 41k에 도시된 바와 같이 끊어지도록 허용한다. 영어의 경우에는, 규칙은 하이픈(4106)을 추가하는 것이다. 그러한 동적으로 삽입된 구두 마크들의 크기 및 배치가 텍스트 언어의 규칙들에 의존할지라도, 통상적으로 임의의 경계 박스 계산에 포함될 것이다.

4. 텍스트의 계산된 모양들은 저장된 경계 박스들이다. 폭이나 높이 또는 이 모두에 대한 제한을 감안했을 때, 텍스트가 얼마나 큰지를 아는 것이 필요한 경우에, 레이아웃 메카니즘에서 이 모양이 사용될 수 있다. 일부 텍스트의 몇몇 가능한 레이아웃들 간에 신속하게 결정하기 위해서, 이러한 모양들은 또한 주어진 차원에서 무엇이 다음으로 크거나 또는 다음으로 작은 모양인지 아는데 사용될 수 있다.

기술된 이와같은 실시예는 공지된 최소 크기의 워드간 공간들을 사용하여, 좌측에서 우측으로 기록된 언어들에 사용되도록 명백히 의도된 것이다. 대안으로, 이러한 규칙들에 따르지 않는 언어들에 대한 배치가 이루어질 수 있다. 예를 들어, 일부 아시아 언어의 경우 세로로 위에서 아래로 글자를 쓰고, 이러한 세로 열을 오른쪽에서 왼쪽으로 적는다. 이런 언어의 경우 수직 및 수평 차원의 역할을 맞바꾸고 글자 사이 공간 크기가 작은 값이나 0으로 줄일 수 있도록 배치 과정을 변경할 수 있다. 따라서 단계 1에서 글자를 수직적으로 스택킹 하고 단계 2에서 높이를 1 단위만큼 줄이며 (글자의 높이가 될 수 있음) 보다 넓은 경계 박스를 만들 수 있도록 과정이 반복된다. 유사하게, 각 언어의 하이픈 규칙들은 텍스트의 언어에 의존할 것이다.

21. 표 생성 예시

도 42a 내지 도 42c는 섹션 18에 기재된 바와 같이 그래프-기초의 레이아웃 메카니즘을 사용하여 표들을 구성하는 몇가지 방법들을 도시한다. 표가 셀들로 나눠진 직사각형 영역으로 정의되며, 각각의 셀은 선택적으로 텍스트 또는 이미지 같은 몇가지 콘텐츠들을 유지한다.

도 42b에 주어진 예에서, 콘테이너들의 상부 열의 하부 에지들은 그들의 하부 에지들이 서로 정확하게 정렬되도록, 0-높이 스트럿들(4206)에 의해 링크된다. 유사하게 콘테이너들의 하부 행의 상부 에지들은 그들이 정렬되도록 0-높이 스트럿에 의해 링크된다. 또한, 이것은 두 행들 간의 가장자리 크기를 정의하고 제어하기 위해 오직 하나의 수직 스트럿(4205)이 필요함을 의미한다. 실제로 도 42a의 예처럼 많은 수의 스트럿을 바꾸는 대신 하나의 스트럿의 선호 또는 실제 거리만 변경하면 되기 때문에 이 사실은 꽤 유용할 수 있다. 가장 좌측 행의 우측 에지들을 링크하고 또한 가장 우측 열의 좌측 에지들을 링크하는 것으로 도시된 폭이 0인 스트럿들(4206)에 대해서도 유사한 논리가 적용된다.

도 42a에서와 같이, 도 42b에서의 예는 페이지에 고정된 표의 외부 에지들을 갖는다. 이런 방식으로 표의 위치와 크기를 고정시키는 대신, 내부 에지들이 배열되는 것과 동일하게, 콘테이너의 모든 외부 에지를 정렬하기 위해 0-길이 스트럿을 이용할 수 있다. 이로 인해, 많은 고정된 외부 에지 위치를 한 개씩 변경할 필요 없이 페이지 상에 필요한 임의의 위치로 표를 이동할 수 있다. 이러한 접근법은 섹션 18.3에 기술된 레이아웃 방법을 사용하여 이루어질 수 있다.

세 번째 대안으로, 표의 외부 에지를 정의하기 위해 가이드를 사용하고, 표의 내부 에지를 정렬하기 위해 0-길이 스트럿들을 사용할 수 있다. 유사한 결과들을 달성하는, 가이드들 및 스트럿들의 많은 다른 실행가능 조합들이 존재한다.

표의 내부 에지를 정렬하는 몇가지 방법이 표의 종래의 개념을 형성하는데 필요함에 유의해야 한다. 도 42c는 내부 에지를 배치하는데 가이드나 스트럿 모두 사용하지 않았을 때 발생할 수 있는 상황을 도시한다. 표의 셀로서 다시 콘테이너(4207)가 사용되며, 근접한 콘테이너들 간의 거리를 정의하기 위해 스트럿(4208)이 사용되었다. 그러나 이러한 예는 열들 및 행들의 내부 에지들이 분리된 채로 유지되지 않기 때문에, 비스듬히 대향된 콘테이너들이 겹치는 것을 방지하는 방법을 제공하지 못한다는 점에 주의해야 한다.

본 발명은 콘테이너 및 콘텐츠의 흥미롭고 실행가능한 배치들을 제시하며, 섹션 18.3 및 그 이후에 기술된 레이아웃 방법은 그러한 배치들을 생성할 수 있다(레이아웃 모델들의 모든 종래 기술의 예들이 가능한 것은 아님). 그러나, 종래의 표의 개념에 대해, 잘 정렬된 열들과 행들을 형성하는데 도 42a 및 도 42b에 도시된 예들이 보다 적합하다.

산업상 이용 가능성

기술된 배치들은 컴퓨터와 데이터 프로세싱 산업들, 특히 필수적으로 동일한 포맷 및 레이아웃을 갖는 대다수의 문서들이 데이터를 변경하면서 재생되는 상황에 적용가능하다. 이러한 예로서, 다른 이름, 주소 및 다른 인적 상세 정보들을 갖는 많은 사람들을 위한 서신의 생성의 경우가 있다. 또 다른 예로서, 다른 휴가 목적지들에 관한 광고책자들의 생성의 경우를 들 수 있는데, 광고책자의 각각은 텍스트와 이미지들을 포함하는 유일한 콘텐츠를 특징으로 하지만, 모든 책자들이 그 광고를 수행하는 조직의 상표 또는 장정(get-up)을 나타내는 공통의 레이아웃에 따라 생성될 수 있다. 많은 다른 예들이 또한 적용될 수 있다.

상술한 것은 본 발명의 단지 몇몇 실시예들만을 기재하고, 본 발명의 범위 및 사상을 벗어나지 않고도 변경들 및/또는 변화들이 행해질 수 있으며, 실시예들은 설명적이지만 제한적인 것은 아니다.

발명의 효과

본 발명에서 제시하는 방법에 따르면, 문서 템플릿을 편집하는 경우에 사용자는 통합된 문서들 중 하나를 항상 보도록 선택할 수 있다. 또한, 콘테이너에 대한 제한들이 같은 위치에 도시되므로, 종래 기술과는 달리 별도의 스크린 또는 영역으로 전환할 필요없이 콘테이너들 상의 여러 위치들을 클릭함으로써 제한들이 그 자리에서 편집될 수 있다. 이는 종래 기술과 비교하여 문서 템플릿의 생성을 간소화한다.

(57) 청구의 범위

청구항 1.

템플릿(template)에 기초하여 가변 데이터 문서(variable dta document)를 위한 레이아웃을 생성하는 방법에 있어서,

상기 레이아웃을 형성하기 위하여 상기 템플릿 내에 적어도 하나의 콘테이너(container)를 세팅하는 단계;

상기 콘테이너의 하나 이상의 특징 중에서 선택된 각각의 특징에 대해 적어도 하나의 제한을 설정하는 단계;

그래픽 사용자 인터페이스(GUI:graphical user interface)를 이용하여, 상기 설정 단계에서 설정된 상기 제한을 상기 세팅 단계에서 세팅된 상기 콘테이너에 시각적으로 디스플레이하는 단계;

콘텐츠를 복수의 콘테이너들에 배치함으로써 상기 문서를 생성하도록 상기 레이아웃을 변경하는 단계

를 포함하고,

상기 레이아웃 내의 상기 적어도 하나의 콘테이너의 적어도 하나의 차원(dimension)과 상기 적어도 하나의 콘테이너의 위치 중의 적어도 하나는, 상기 레이아웃 내의 각각의 제한이 만족되는 것을 조건으로 상기 배치된 콘텐츠의 속성에 기초하여 변경되는 레이아웃 생성 방법.

청구항 2.

제1항에 있어서,

상기 설정 단계는 상기 콘테이너의 각각의 특징에 대해 두 개의 상기 제한들을 각각 설정하는 단계를 포함하는 레이아웃 생성 방법.

청구항 3.

제2항에 있어서,

상기 특징들은 상기 콘테이너의 코너(corner)들, 상기 콘테이너의 측변들(sides)의 중심점(centre-point)들, 상기 측면들의 비중심점(non-centre-point)들, 상기 콘테이너의 중심라인(centre-line)들 상의 비중심점들, 및 상기 콘테이너의 중심점으로 구성된 그룹으로부터 선택되는 레이아웃 생성 방법.

청구항 4.

제1항에 있어서,

상기 배치된 콘텐츠의 상기 속성은 상기 배치된 콘텐츠의 크기(size)를 포함하는 레이아웃 생성 방법.

청구항 5.

제4항에 있어서,

상기 크기는 상기 배치된 콘텐츠의 폭을 포함하는 레이아웃 생성 방법.

청구항 6.

제4항에 있어서,

상기 크기는 상기 배치된 콘텐츠의 높이를 포함하는 레이아웃 생성 방법.

청구항 7.

제1항에 있어서,

사용자 유발 선택(user instigated selection)은, 그래픽 사용자 인터페이스 내의 상기 레이아웃의 표현에 의해 묘사된 상기 대응하는 특징에 일치시킨 포인팅 디바이스를 사용자가 조작(actuation)함으로써 수행되는 레이아웃 생성 방법.

청구항 8.

제1항에 있어서, 상기 설정 단계는

이전에 어떠한 제한도 설정되지 않았던 경우 상기 대응하는 특징에 제한을 설정하는 단계; 및

이전에 제한이 설정되었던 경우 상기 제한을 제거하기 위하여 상기 대응하는 특징을 리셋팅하는 단계

중의 하나를 포함하는 레이아웃 생성 방법.

청구항 9.

제8항에 있어서,

설정된 제한을 갖는 상기 각각의 특징에 제한 심볼을 제공하는 레이아웃 생성 방법.

청구항 10.

제1항에 있어서,

상기 콘테이너는 직사각형이고, 상기 차원들은 상기 콘테이너의 폭과 높이를 포함하며, 상기 콘텐츠는 상기 콘테이너 각각에 대해 텍스트 및/또는 이미지 콘텐츠를 포함하는 레이아웃 생성 방법.

청구항 11.

제10항에 있어서,

상기 특징들은 상기 콘테이너의 코너들을 포함하고, 설정된 제한을 갖는 각각의 상기 코너는 상기 레이아웃에 고정된 채로 유지되는 레이아웃 생성 방법.

청구항 12.

제10항에 있어서,

상기 특징들은 상기 콘테이너의 에지들의 중심점들을 포함하고, 설정된 제한을 갖는 각각의 상기 에지 중심에 대해, (1) 상기 에지가 대칭적으로 확장 또는 수축할 수 있고, (2) 상기 에지의 상기 중심이 고정된 채로 유지되는 조건에 기초하여 상기 콘테이너의 상기 차원들 및/또는 위치가 변경되는 레이아웃 생성 방법.

청구항 13.

제10항에 있어서,

상기 특징들은 상기 콘테이너의 대응하는 에지들 상에 비중심점들을 포함하고, 설정된 제한을 갖는 각각의 에지에 대해, (1) 상기 에지의 길이가 자유롭게 확장 또는 수축할 수 있고, (2) 상기 에지는 상기 에지에 수직인 방향으로 이동하지 않는 것을 조건으로 상기 콘테이너의 상기 차원들 및/또는 위치가 변경되는 레이아웃 생성 방법.

청구항 14.

제10항에 있어서,

상기 특징들은 상기 콘테이너의 수평 및 수직 중심라인들 상의 비중심 위치들(non-centre locations)을 포함하고, 설정된 제한을 갖는 각각의 중심라인에 대해, (1) 상기 중심라인은 고정된 채로 유지되고, (2) 상기 중심라인에 평행한 에지들은 대칭적으로 확장 또는 수축할 수 있고, (3) 상기 중심라인에 수직인 에지들은 상기 고정된 중심라인과 독립적으로 위치될 수 있는 것을 조건으로 상기 콘테이너의 상기 차원들 및 위치 중 적어도 하나가 변경되는 레이아웃 생성 방법.

청구항 15.

제10항에 있어서,

상기 하나의 특징은 상기 콘테이너의 중심점을 포함하고, 설정된 제한을 갖는 중심점에 대해, (1) 상기 중심점이 고정된 채로 유지되고, (2) 상기 콘테이너의 제1 쌍의 평행 에지들이 제1의 양만큼 확장 또는 수축하고, 평행 에지들의 다른 쌍이 제2의 양만큼 확장 또는 수축하는 것을 조건으로 상기 콘테이너의 상기 차원들 및/또는 위치가 변경되는 레이아웃 생성 방법.

청구항 16.

제11항에 있어서,

상기 콘테이너의 다른 코너에 추가적인 제한을 설정하는 단계를 더 포함하고, 상기 추가적인 제한은 소정의 규칙에 따라서 설정되는 레이아웃 생성 방법.

청구항 17.

제1항에 있어서,

상기 세팅 단계는, 상기 레이아웃을 형성하기 위해 상기 템플릿 내에 제1 및 제2 콘테이너들을 세팅하는 단계를 포함하고;

상기 설정 단계는 상기 제1 콘테이너의 제1 에지를 선택하는 단계 및 상기 제2 콘테이너의 제2 에지를 선택하는 단계를 포함하며 - 상기 선택 단계는 상기 GUI를 이용하여 상기 제1 및 제2 에지들의 선택을 디스플레이하는 단계를 포함함 - ;

상기 변경 단계는, 상기 콘테이너들에 콘텐츠를 배치함으로써 상기 문서를 생성하도록 상기 레이아웃을 변경하는 단계를 포함하는 - 상기 제1 콘테이너 및 상기 제2 콘테이너의 차원 및 위치 중의 적어도 하나는, 상기 제1 에지 및 상기 제2 에지 간의 위치적 오프셋(positional offset)이 고정되어 있는 것을 조건으로 상기 배치된 콘텐츠의 속성에 기초하여 변경됨 -

레이아웃 생성 방법.

청구항 18.

제17항에 있어서,

상기 사용자 유발 선택은, 그래픽 사용자 인터페이스 내의 상기 레이아웃의 표현에 의해 묘사된 상기 대응하는 에지에 적어도 맞춘 포인팅 디바이스를 사용자가 조작함으로써 수행되고, 상기 방법은 상기 제1 에지와 상기 제2 에지 간의 상기 위치적 오프셋을 표시하는 스트럿 심볼(strut symbol)을 상기 그래픽 사용자 인터페이스에 디스플레이하는 단계를 더 포함하는 레이아웃 생성 방법.

청구항 19.

제18항에 있어서,

상기 선택은, 상기 제1 콘테이너 내에서 상기 포인팅 디바이스를 선택적으로 조작하여 상기 조작된 포인팅 디바이스를 상기 제2 콘테이너로 이동시키고 이에 의해 상기 제1 및 제2 에지들을 확장함으로써 수행되고, 상기 포인팅 디바이스의 선택 해제(deselective) 조작으로 상기 에지들 사이에 상기 스트럿을 형성하는 레이아웃 생성 방법.

청구항 20.

제18항에 있어서,

상기 선택은, 상기 제1 콘테이너 내에서의 상기 포인팅 디바이스의 제1 조작 및 상기 제2 콘테이너 내에서의 상기 포인팅 디바이스의 상기 제2 조작에 의해 수행되는 레이아웃 생성 방법.

청구항 21.

제1항에 있어서,

상기 설정 단계는 상기 레이아웃 내에 가이드를 설정하는 단계 및 상기 하나의 콘테이너와 상기 가이드 사이에 오프셋을 세팅하는 단계를 포함하고;

상기 변경 단계는 상기 하나의 콘테이너 내에 콘텐츠를 배치하여 상기 문서를 생성함으로써 상기 레이아웃을 변경하는 단계를 포함하는 - 상기 오프셋이 유지되는 것을 조건으로 상기 하나의 콘테이너의 차원 및 위치 중의 적어도 하나가 상기 배치된 콘텐츠의 속성에 기초하여 변경됨 - 레이아웃 생성 방법.

청구항 22.

제21항에 있어서,

상기 템플릿 내에 추가적인 콘테이너를 세팅하는 단계 및 상기 추가적인 콘테이너와 상기 가이드 사이에 추가적인 오프셋을 세팅하는 단계를 더 포함하는 레이아웃 생성 방법.

청구항 23.

제21항에 있어서, 상기 레이아웃 내에서 상기 가이드를 이동시키는 단계를 더 포함하는 레이아웃 생성 방법.

청구항 24.

제21항에 있어서, 상기 오프셋의 상기 세팅 단계는, 그래픽 사용자 인터페이스 내에서 상기 레이아웃의 표현에 의해 묘사되는 상기 가이드에 맞춘 포인팅 디바이스를 사용자가 조작함으로써 수행되며, 상기 방법은 상기 가이드와 상기 콘테이너 간의 연관성을 나타내는 연관 심볼(association symbol)을 상기 그래픽 사용자 인터페이스에 디스플레이하는 단계를 더 포함하는 레이아웃 생성 방법.

청구항 25.

제1항에 있어서,

상기 세팅 단계는 상기 레이아웃을 형성하기 위해 상기 템플릿 내에 제1 콘테이너 및 제2 콘테이너를 세팅하는 단계를 포함하고;

상기 변경 단계는 상기 제1 콘테이너 및 상기 제2 콘테이너 내에 콘텐츠를 배치하여 상기 문서를 생성함으로써 상기 레이아웃을 변경하는 단계를 포함하며, 상기 제1 콘테이너의 적어도 하나의 속성이 변경되어 상기 제2 콘테이너의 대응하는 하나의 속성과 동기화되는 것을 조건으로, 상기 제1 콘테이너 및 상기 제2 콘테이너의 차원 및 위치 중 적어도 하나가 상기 콘텐츠의 속성에 기초하여 변경되는 레이아웃 생성 방법.

청구항 26.

제25항에 있어서,

상기 콘텐츠는 적어도 텍스트 콘텐츠를 포함하고, 상기 제1 콘테이너의 상기 하나의 속성은 상기 텍스트 콘텐츠의 텍스트 크기, 라인 간격(line spacing) 및 문자 간격(character spacing)으로 이루어진 그룹으로부터 선택되는 레이아웃 생성 방법.

청구항 27.

제1항에 있어서,

상기 설정 단계는 인접한 열들이 거터(gutter)에 의해 분리되도록 상기 적어도 하나의 콘테이너 내에 복수의 열들을 설정하는 단계를 더 포함하고;

상기 변경 단계는, 상기 거터들의 차원들은 고정된 반면, 상기 하나의 콘테이너의 차원들 및 상기 열들은 가변적이도록 상기 적어도 하나의 콘테이너에 텍스트 콘텐츠를 배치함으로써 상기 문서를 변경하는 단계를 포함하는 레이아웃 생성 방법.

청구항 28.

제27항에 있어서,

상기 콘테이너는 시각적 배경을 가지며, 상기 다수의 열들을 설정하는 상기 단계는 상기 열들이 위치되는 상기 콘테이너 내에 경계를 설정하는 단계를 더 포함하는 레이아웃 생성 방법.

청구항 29.

제1항에 있어서,

상기 설정 단계는 인접하는 열들이 거터에 의해 분리되도록 적어도 하나의 상기 콘테이너 내에 복수의 열들을 설정하는 단계를 더 포함하고;

상기 변경 단계는, 상기 거터들의 차원들은 고정된 반면 상기 하나의 콘테이너 및 상기 열들의 차원들이 가변적이도록, 상기 하나의 콘테이너에 적어도 텍스트 콘텐츠를 배치함으로써 적어도 하나의 상기 문서를 생성하기 위해 상기 레이아웃을 변경하는 단계를 포함하는 레이아웃 생성 방법.

청구항 30.

제1항에 있어서,

상기 설정 단계는 상기 텍스트가 배치될 인접한 열들이 두 개의 경계들 및 상기 두 개의 경계들 사이의 중심에 위치하는 거터 라인에 의해 형성되도록 상기 적어도 하나의 콘테이너 내에 다수의 열들을 설정하는 단계와, 인접한 상기 열들 간의 거터의 차원들을 변경시키기 위해 상기 두개의 경계들 중 하나의 위치를 변경 시키는 단계를 더 포함하고;

상기 변경 단계는 상기 적어도 하나의 콘테이너에 텍스트 콘텐츠를 배치함으로써 상기 문서를 변경하는 단계를 포함하는 레이아웃 생성 방법.

청구항 31.

제30항에 있어서,

상기 하나의 경계의 상기 위치는 상기 하나의 경계를 따르는 위치에서의 상기 하나의 경계에 대한 사용자 유발 선택에 의해 변경되는 레이아웃 생성 방법.

청구항 32.

제31항에 있어서,

상기 선택은, 그래픽 사용자 인터페이스 내의 상기 레이아웃의 표현에 의해 묘사된 상기 하나의 경계에 맞춘 포인팅 디바이스를 사용자가 조작하여 상기 레이아웃 내의 다른 위치로 상기 선택된 경계를 이동시킴으로써 수행되는 레이아웃 생성 방법.

청구항 33.

제1항에 있어서,

상기 변경 단계는, 상기 콘테이너의 내부 가장자리가 상기 콘테이너의 배경색 및 상기 템플릿의 배경 간의 관계에 기초하여 결정된다는 조건에 기초하여 상기 콘테이너의 차원 및 위치 중 적어도 하나가 변경되도록, 상기 콘테이너에 콘텐츠를 배치하여 상기 문서를 생성함으로써 상기 레이아웃을 변경하는 단계를 포함하는 레이아웃 생성 방법.

청구항 34.

제1항에 있어서,

상기 설정 단계는 상기 콘테이너와 연관된 고정된 크기 및 가변적인 위치를 갖는 콘테이너 범위를 설정하는 단계를 포함하고;

상기 변경 단계는, 상기 콘태이너가 상기 연관된 콘테이너 범위를 벗어나지 않는 것을 조건으로 상기 콘테이너의 차원 및 위치 중 적어도 하나가 상기 배치된 콘텐츠의 속성에 기초하여 변경되도록, 상기 콘테이너에 콘텐츠를 배치하여 상기 문서들을 생성함으로써 상기 레이아웃을 변경하는 단계를 포함하는 레이아웃 생성 방법.

청구항 35.

제34항에 있어서,

상기 콘테이너 범위는 상기 콘테이너를 둘러싸는 최대 범위인 레이아웃 생성 방법.

청구항 36.

제34항에 있어서,

상기 콘테이너 범위는 상기 콘테이너 내의 최소 범위인 레이아웃 생성 방법.

청구항 37.

제1항에 있어서,

상기 적어도 하나의 콘테이너는 높이 및 폭을 가지고;

상기 설정 단계는 상기 높이 및 폭 중 적어도 하나를 고정된 차원으로 세팅하는 단계를 포함하며;

상기 변경 단계는, 상기 높이 및 폭 중 하나의 세팅 값이 상기 고정된 차원인 채로 유지되는 것을 조건으로 상기 콘테이너의 추가적인 차원 및 위치 중의 적어도 하나가 상기 배치된 콘텐츠의 속성에 기초하여 변경되도록, 상기 콘테이너에 콘텐츠를 배치하여 상기 문서들을 생성함으로써 상기 레이아웃을 변경하는 단계를 포함하는 레이아웃 생성 방법.

청구항 38.

제37항에 있어서,

상기 세팅 단계는 상기 하나의 높이 및 폭을 정의하는, 상기 콘테이너의 에지들에 대한 사용자 유발 선택을 검출하는 단계를 포함하는 레이아웃 생성 방법.

청구항 39.

제38항에 있어서,

상기 사용자 유발 선택은 그래픽 사용자 인터페이스 내의 상기 레이아웃의 표현에 의해 묘사되는 상기 대응하는 에지에 적어도 일치시킨 포인팅 디바이스를 사용자가 조작함으로써 수행되며, 상기 방법은 상기 에지들 사이의 고정된 차원을 나타내는 심볼을 상기 그래픽 사용자 인터페이스에 디스플레이하는 단계를 더 포함하는 레이아웃 생성 방법.

청구항 40.

제1항에 있어서,

상기 세팅 단계는 상기 레이아웃을 형성하기 위해 상기 템플릿 내에 콘테이너들의 어레이를 세팅하는 단계를 포함하고;

상기 설정 단계는 (i) 상기 콘테이너들에 의해 정의된 셀들을 형성하는 열들 및 행들로 상기 어레이를 분리하기 위해 상기 레이아웃에 복수의 가이드들을 세팅하는 단계 및, (ii) 상기 각각의 가이드와 상기 각각의 콘테이너의 대응하는 인접 측변 사이에 오프셋을 설정하는 단계를 포함하며;

상기 변경 단계는, 상기 오프셋들이 유지되는 것을 조건으로 상기 배치된 콘텐츠의 속성에 기초하여 상기 콘테이너들의 차원 및 위치 중의 적어도 하나가 변경되도록, 상기 콘테이너들에 콘텐츠를 배치하여 상기 문서를 생성함으로써 상기 레이아웃을 변경하는 단계를 포함하는 레이아웃 생성 방법.

청구항 41.

제1항에 있어서,

상기 세팅 단계는 상기 레이아웃을 형성하기 위해 상기 템플릿 내의 열들 및 행들로 콘테이너들의 어레이를 세팅하는 단계를 포함하고;

상기 설정 단계는 상기 열들 및 행들의 마진 크기를 지정하기 위해 인접한 콘테이너들의 에지들 사이에 오프셋들을 설정하는 단계를 포함하며;

상기 변경 단계는, 상기 오프셋들이 유지되는 것을 조건으로 상기 배치된 콘텐츠의 속성에 기초하여 상기 콘테이너들의 차원 및 위치 중의 적어도 하나가 변경되도록, 상기 콘테이너들에 콘텐츠를 배치하여 상기 문서를 생성함으로써 상기 레이아웃을 변경하는 단계를 포함하는 레이아웃 생성 방법.

청구항 42.

제41항에 있어서,

상기 오프셋들은 상기 인접한 열 및 행 내의 제1 콘테이너 및 바로 인접한 콘테이너들 사이에 세팅되고, 상기 방법은 각각의 열 및 행 내의 인접한 콘테이너들 간의 복수의 0-길이(zero-length) 오프셋들을 세팅하는 단계를 더 포함하는 레이아웃 생성 방법.

청구항 43.

프로그램을 수록하고, 템플릿에 기초하여 가변 데이터 문서를 위한 레이아웃을 생성하기 위한 절차를 컴퓨터가 실행하도록 적응된 컴퓨터 판독가능 기록매체에 있어서, 상기 프로그램은:

상기 레이아웃을 형성하기 위하여 상기 템플릿 내에 적어도 하나의 콘테이너를 세팅하기 위한 코드;

상기 콘테이너의 적어도 하나의 특징 중 선택된 각각의 특징에 대해 적어도 하나의 제한을 설정하기 위한 코드;

그래픽 사용자 인터페이스(GUI)를 이용하여 상기 설정된 제한을 상기 세팅된 콘테이너에 시각적으로 디스플레이하기 위한 코드; 및

콘텐츠를 복수의 콘테이너들에 배치함으로써 상기 문서를 생성하도록 상기 레이아웃을 변경하기 위한 코드

를 포함하고, 상기 적어도 하나의 콘테이너의 적어도 하나의 차원과 상기 레이아웃 내의 상기 적어도 하나의 콘테이너의 위치 중 적어도 하나는, 상기 레이아웃 내의 각각의 제한이 만족되는 것을 조건으로 상기 배치된 콘텐츠의 속성에 기초하여 변경되는 컴퓨터 판독가능 기록매체.

청구항 44.

템플릿에 기초하여 가변 데이터 문서를 위한 레이아웃을 생성하기 위한 컴퓨터 장치에 있어서,

상기 레이아웃을 형성하기 위하여 상기 템플릿 내에 적어도 하나의 콘테이너를 세팅하기 위한 수단;

상기 콘테이너의 적어도 하나의 특징 중 선택된 각각의 특징에 대해 적어도 하나의 제한을 설정하기 위한 수단;

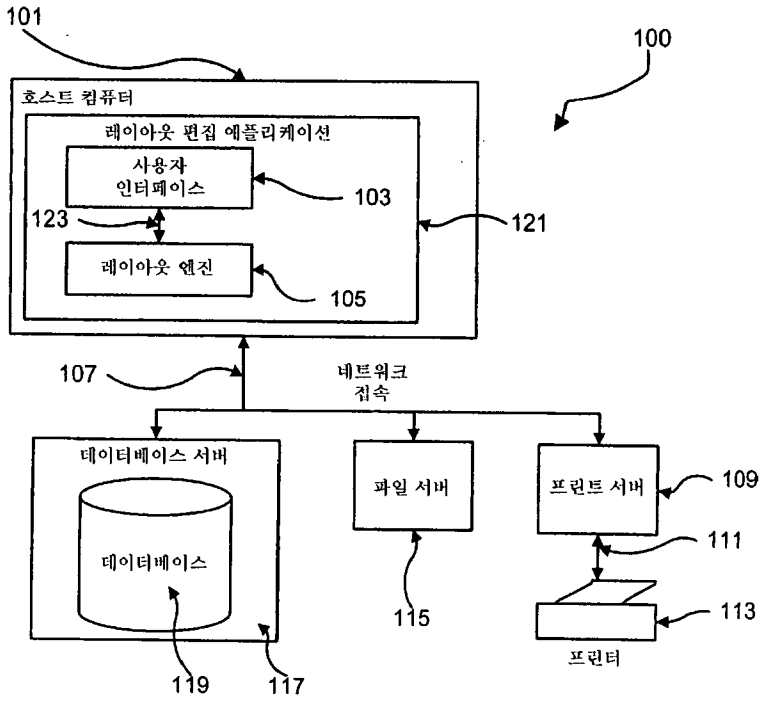
디스플레이 디바이스 상에서 그래픽 사용자 인터페이스(GUI)를 이용하여 상기 설정된 제한을 상기 세팅된 콘테이너에 시각적으로 디스플레이하기 위한 수단; 및

콘텐츠를 복수의 콘테이너들에 배치함으로써 상기 문서를 생성하도록 상기 레이아웃을 변경하기 위한 수단

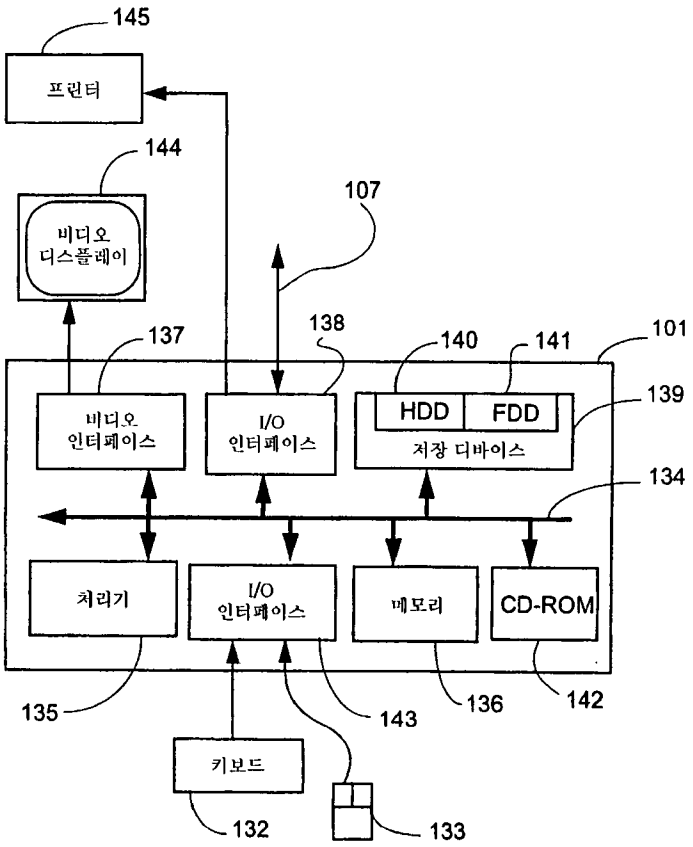
을 포함하고, 상기 적어도 하나의 콘테이너의 적어도 하나의 차원과 상기 레이아웃 내의 상기 적어도 하나의 콘테이너의 위치 중 적어도 하나는, 상기 레이아웃 내의 각각의 제한이 만족되는 것을 조건으로 상기 배치된 콘텐츠의 속성에 기초하여 변경되는 컴퓨터 장치.

도면

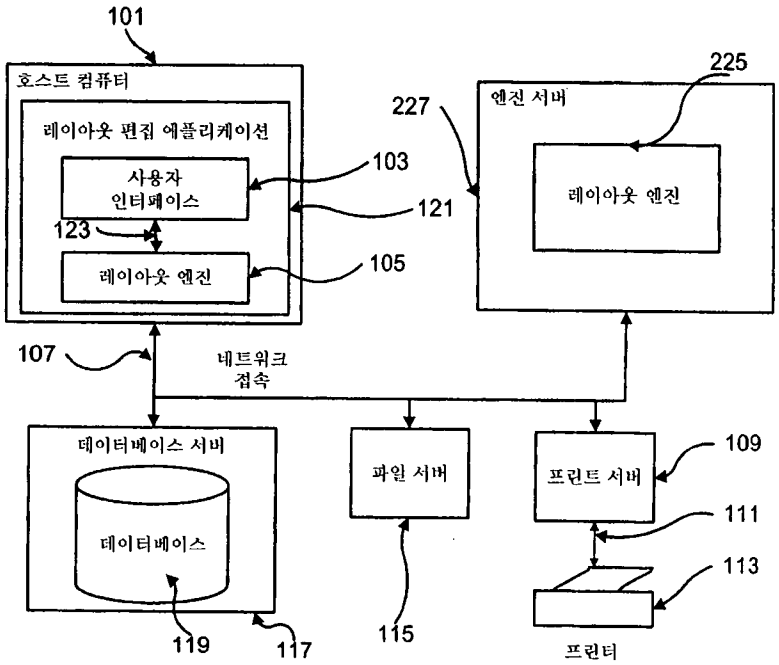
도면1a



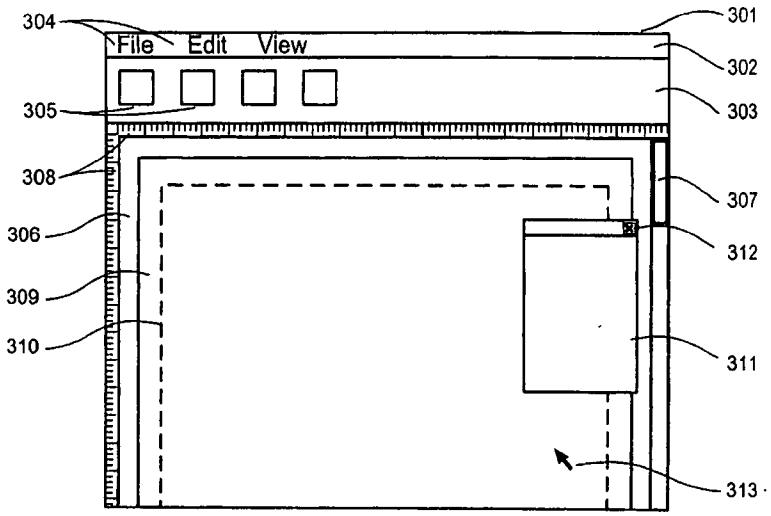
도면1b


145
프린터
144
비디오 디스플레이
107
137
138
140
141
101
비디오 인터페이스
I/O 인터페이스
HDD
FDD
저장 디바이스
139
134
처리기
I/O 인터페이스
메모리
CD-ROM
135
143
136
142
키보드
132
133

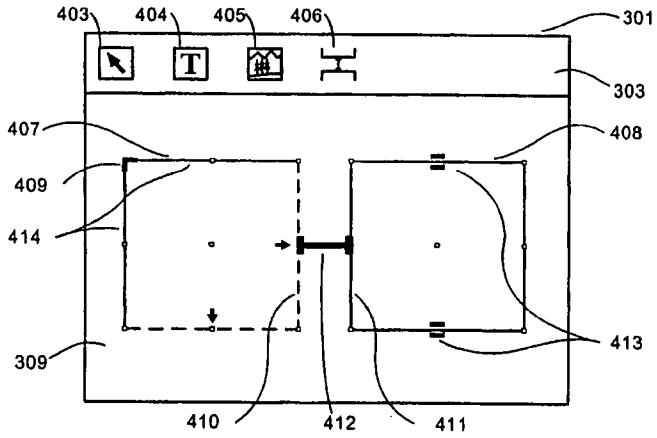
도면2


101
호스트 컴퓨터
레이아웃 편집 애플리케이션
사용자 인터페이스
123
레이아웃 엔진
103
121
105
225
엔진 서버
227
레이아웃 엔진
107
네트워크 접속
데이터베이스 서버
데이터베이스
119
117
파일 서버
115
프린트 서버
109
111
113
프린터

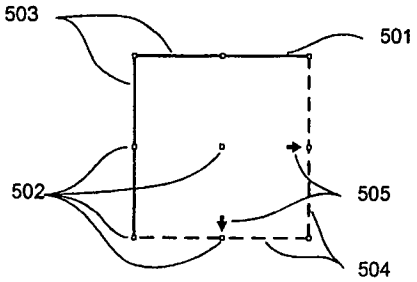
도면3


304
File
Edit
View
301
302
303
305
308
307
306
312
309
310
311
313

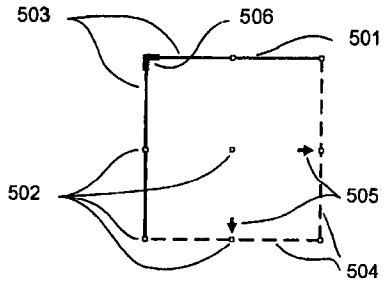
도면4


403
404
405
406
301
303
407
408
409
414
413
309
410
412
411

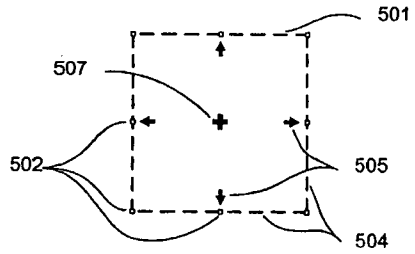
도면5a


503
501
502
505
504

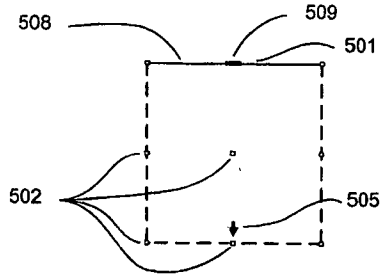
도면5b



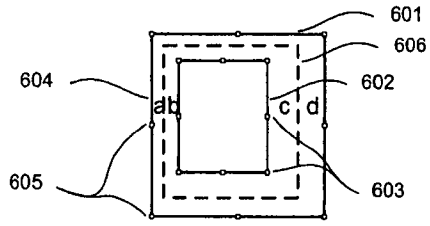
도면5c



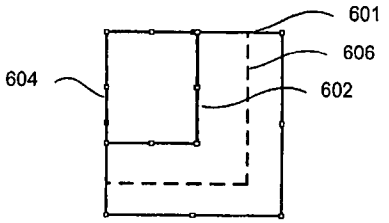
도면5d



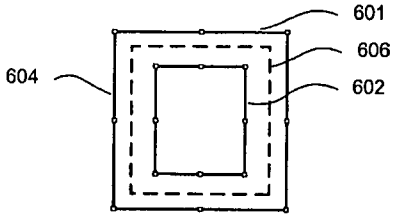
도면6a



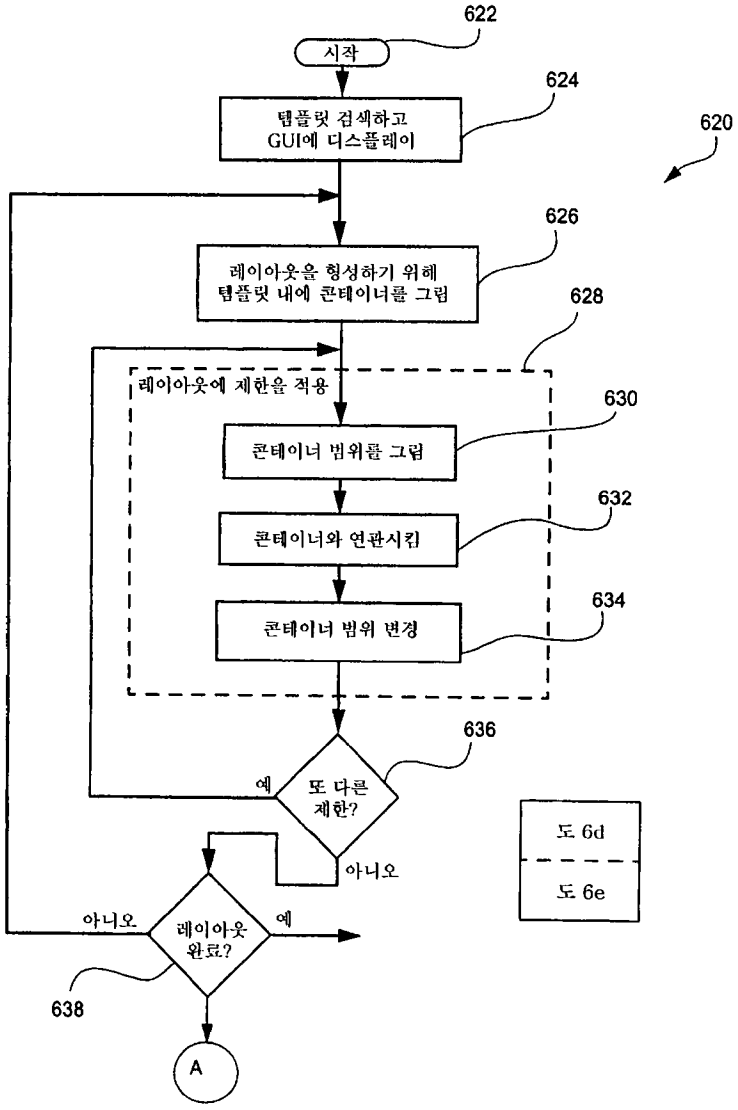
도면6b



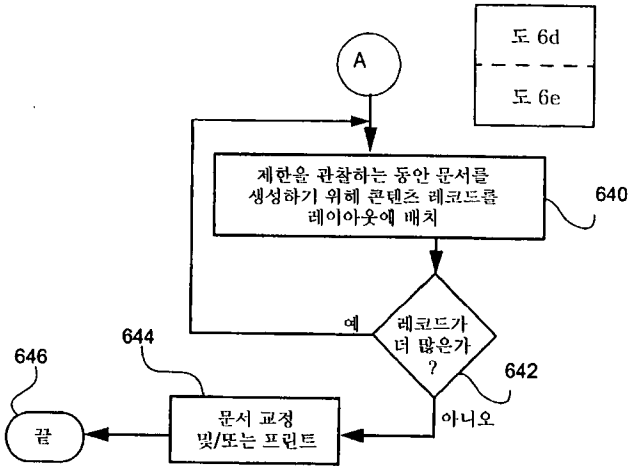
도면6c



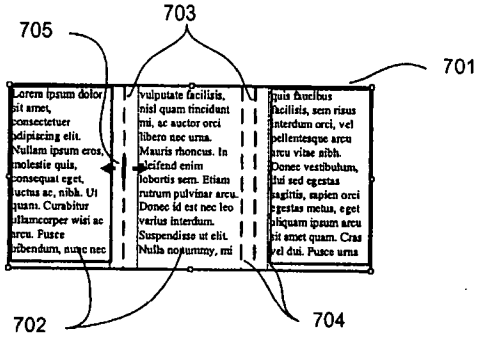
도면6d


622
시작
624
템플릿 검색하고
GUI에 디스플레이
620
626
레이아웃을 형성하기 위해
템플릿 내에 콘테이너를 그림
628
레이아웃에 제한을 적용
630
콘테이너 범위를 그림
632
콘테이너와 연관시킴
634
콘테이너 범위 변경
636
예
또 다른
제한?
아니오
아니오
레이아웃
완료?
예
638
A
도 6d
도 6e

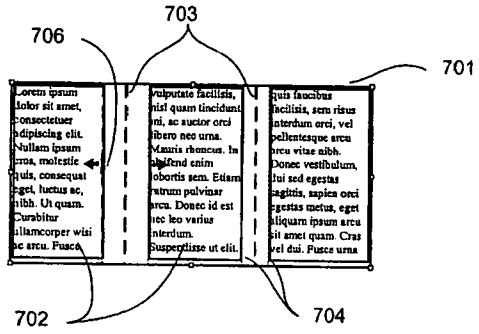
도면6e



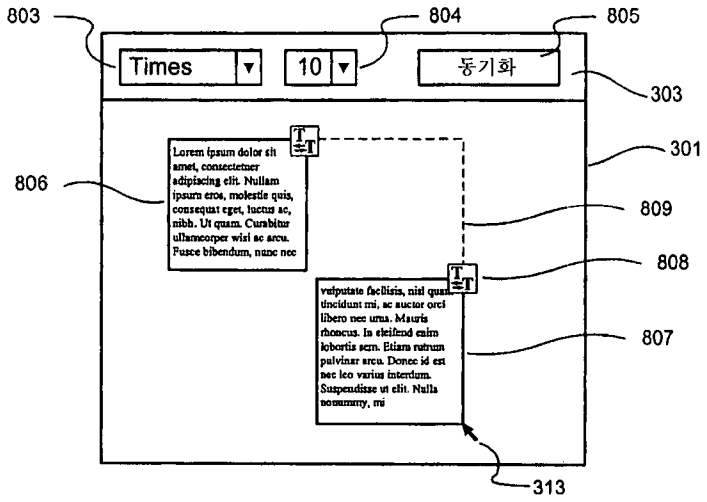
도면7a



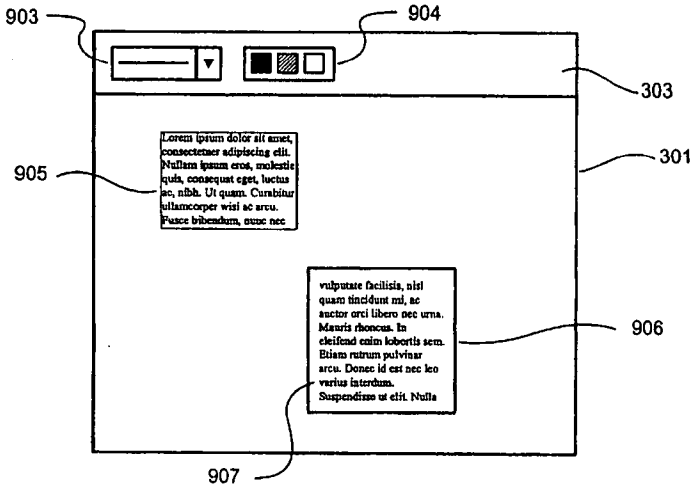
도면7b



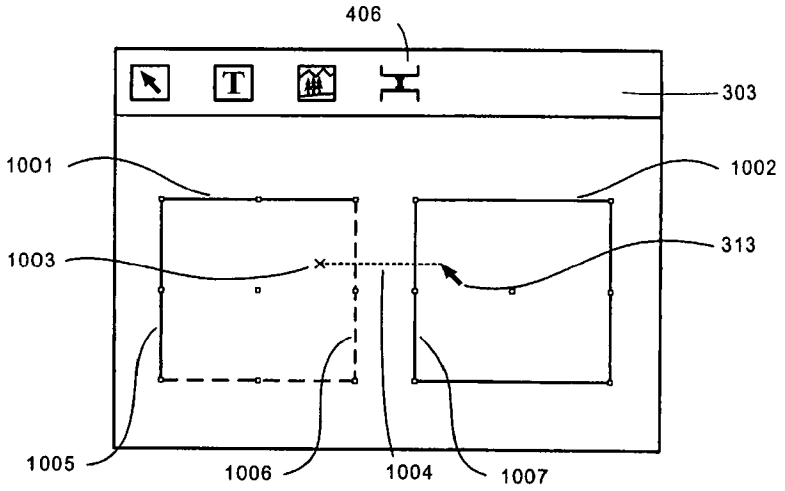
도면8


803
804
805
Times
10
동기화
303
301
806
809
808
807
313

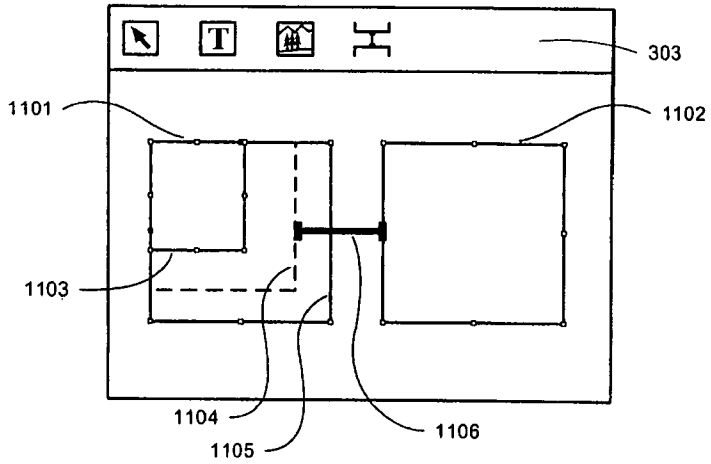
도면9


903
904
303
905
301
906
907

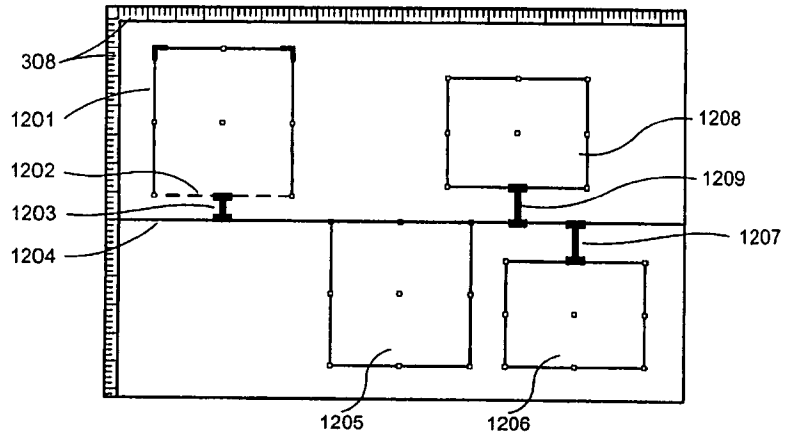
도면10



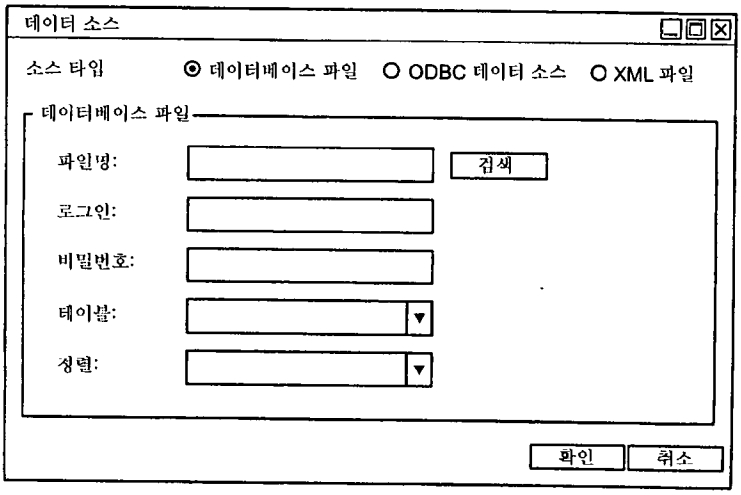
도면11



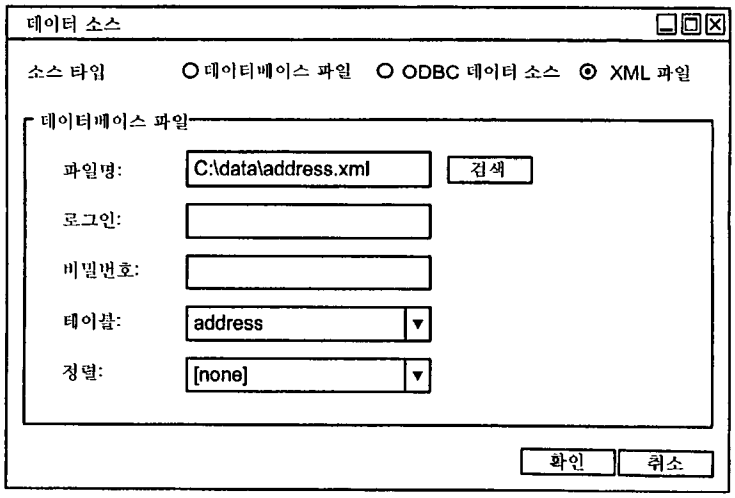
도면12



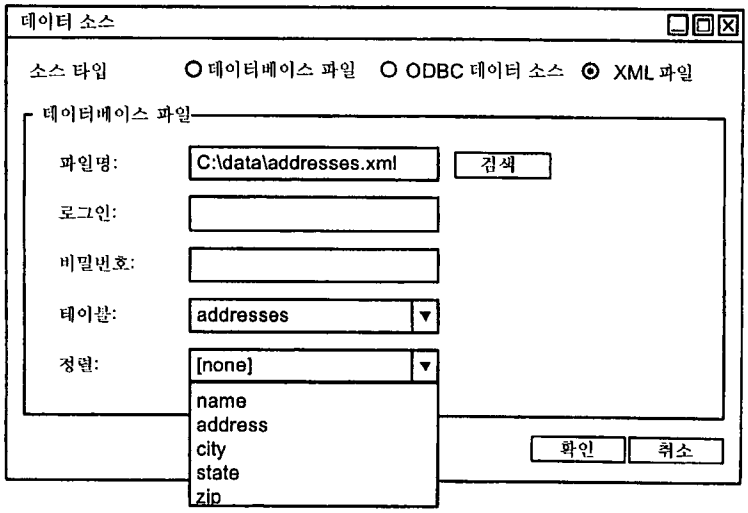
도면13


데이터 소스
소스 타입
데이터베이스 파일
ODBC 데이터 소스
XML 파일
데이터베이스 파일
파일명:
검색
로그인:
비밀번호:
테이블:
정렬:
확인
취소

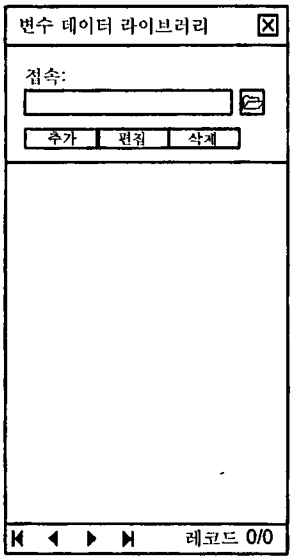
도면14


데이터 소스
소스 타입
데이터베이스 파일
ODBC 데이터 소스
XML 파일
데이터베이스 파일
파일명:
C:\data\address.xml
검색
로그인:
비밀번호:
테이블:
address
정렬:
[none]
확인
취소

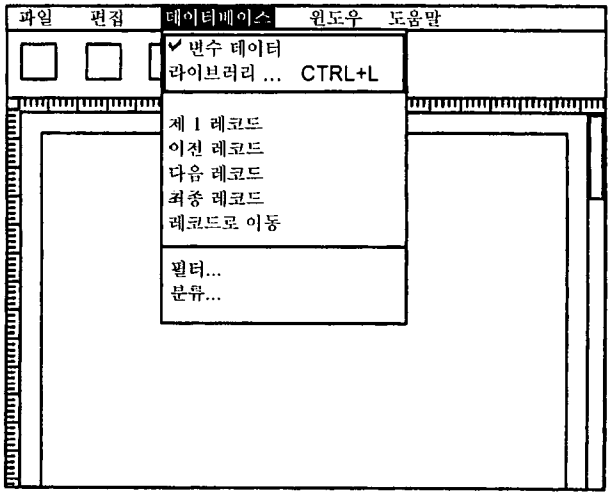
도면15


데이터 소스
소스 타입
데이터베이스 파일
ODBC 데이터 소스
XML 파일
데이터베이스 파일
파일명:
C:\data\addresses.xml
검색
로그인:
비밀번호:
테이블:
addresses
정렬:
[none]
name
address
city
state
zip
확인
취소

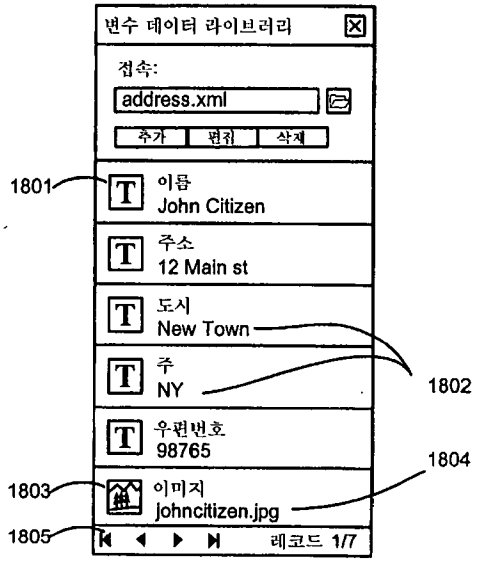
도면16


변수 데이터 라이브러리
접속:
추가
편집
삭제
레코드 0/0

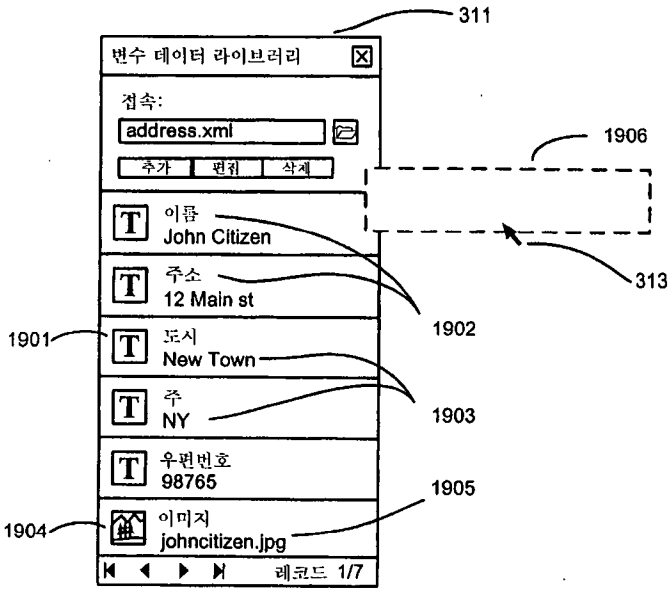
도면17


파일
편집
데이터베이스
윈도우
도움말
✔ 변수 데이터
라이브러리 ... CTRL+L
제 1 레코드
이전 레코드
다음 레코드
최종 레코드
레코드로 이동
필터...
분류...

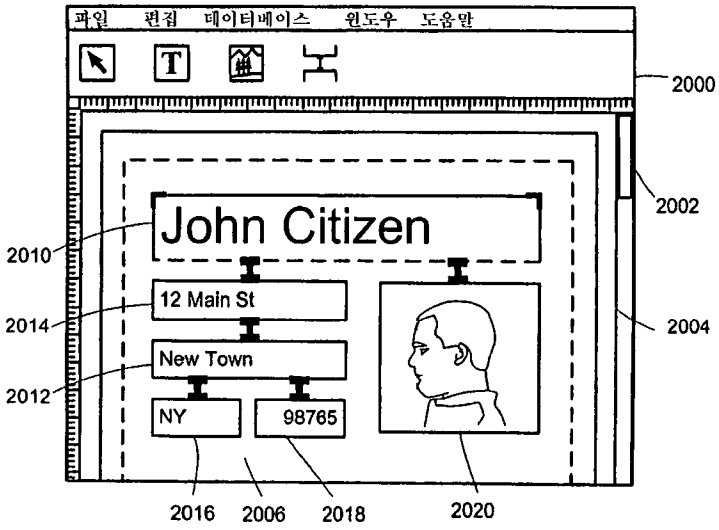
도면18


변수 데이터 라이브러리
접속:
address.xml
추가
편집
삭제
1801
이름
John Citizen
주소
12 Main st
도시
New Town
주
NY
1802
우편번호
98765
1804
1803
이미지
johncitizen.jpg
1805
레코드 1/7

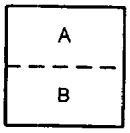
도면19



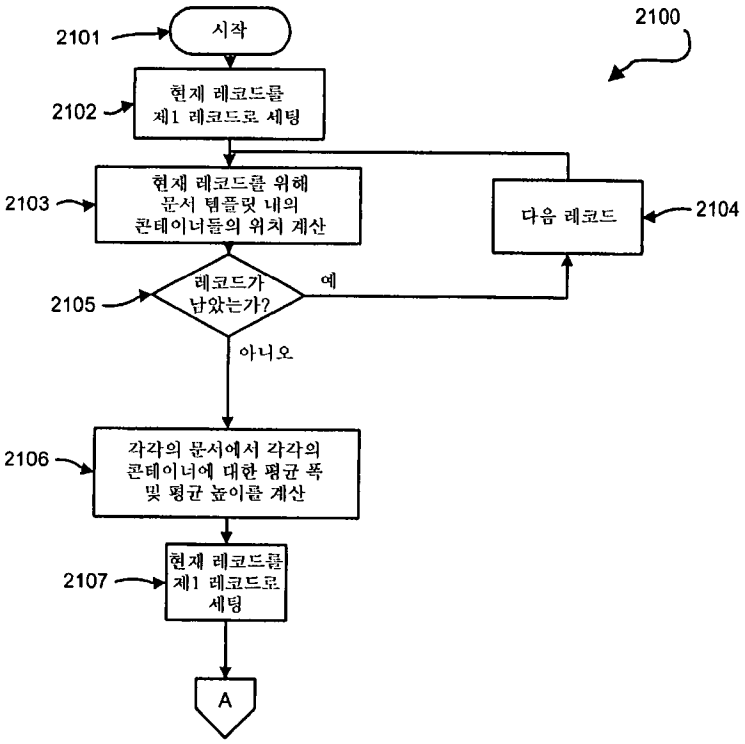
도면20



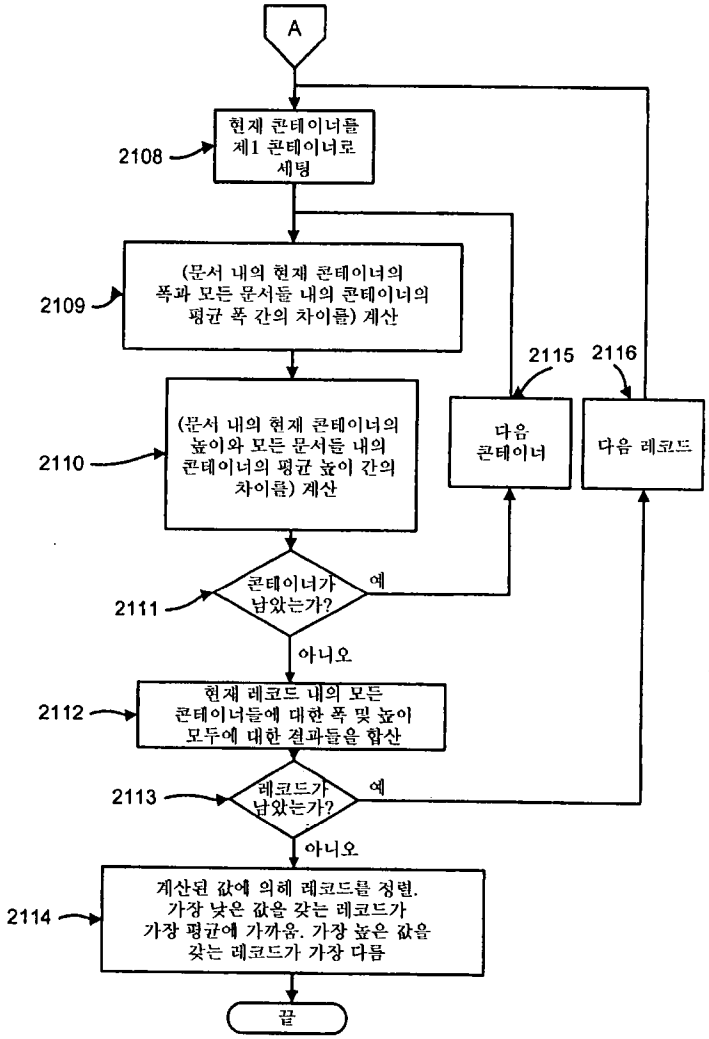
도면21



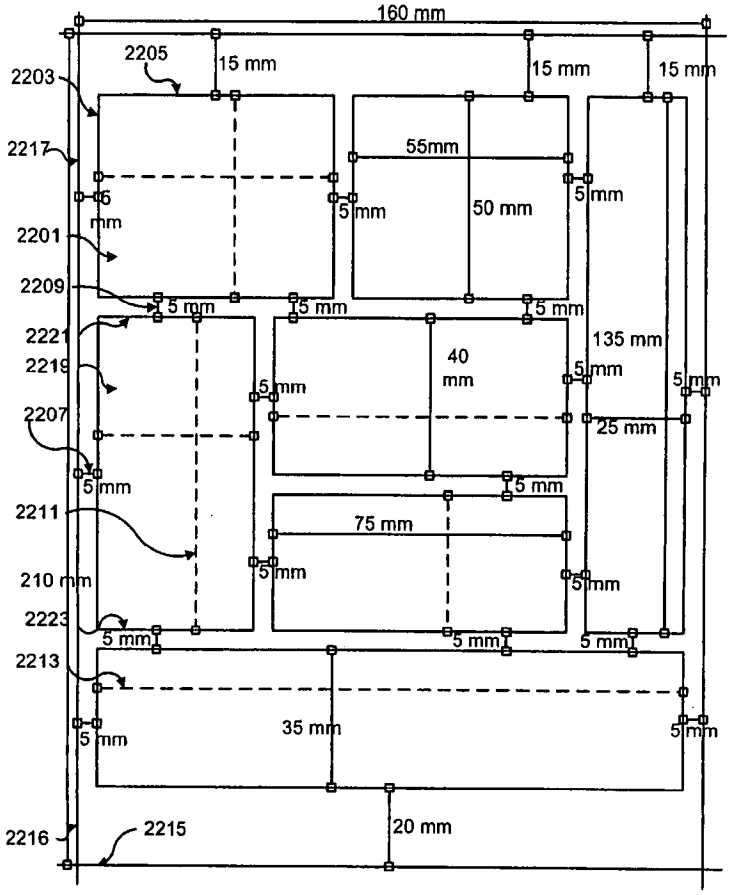
도면21a


2100
2101
시작
2102
현재 레코드를 제1 레코드로 세팅
2103
현재 레코드를 위해 문서 템플릿 내의 콘테이너들의 위치 계산
2104
다음 레코드
2105
레코드가 남았는가?
예
아니오
2106
각각의 문서에서 각각의 콘테이너에 대한 평균 폭 및 평균 높이를 계산
2107
현재 레코드를 제1 레코드로 세팅
A

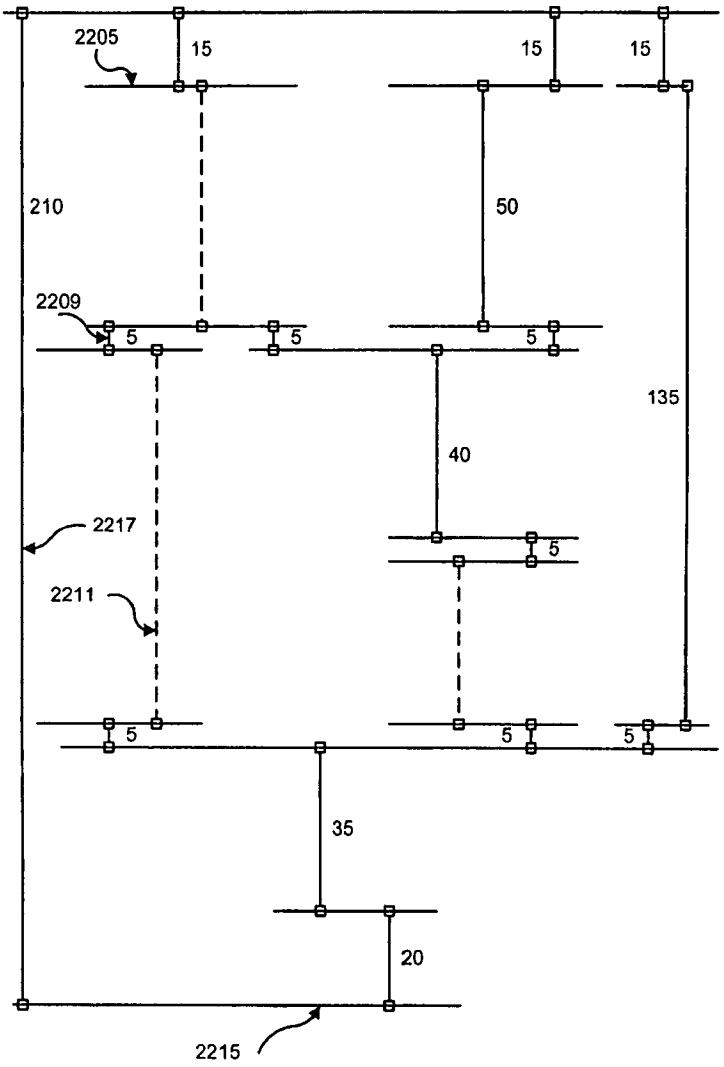
도면21b


A
현재 콘테이너를 제1 콘테이너로 세팅
2108
(문서 내의 현재 콘테이너의 폭과 모든 문서들 내의 콘테이너의 평균 폭 간의 차이를) 계산
2109
(문서 내의 현재 콘테이너의 높이와 모든 문서들 내의 콘테이너의 평균 높이 간의 차이를) 계산
2110
콘테이너가 남았는가?
2111
예
아니오
다음 콘테이너
2115
다음 레코드
2116
현재 레코드 내의 모든 콘테이너들에 대한 폭 및 높이 모두에 대한 결과들을 합산
2112
레코드가 남았는가?
2113
예
아니오
계산된 값에 의해 레코드를 정렬. 가장 낮은 값을 갖는 레코드가 가장 평균에 가까움. 가장 높은 값을 갖는 레코드가 가장 다름
2114
끝

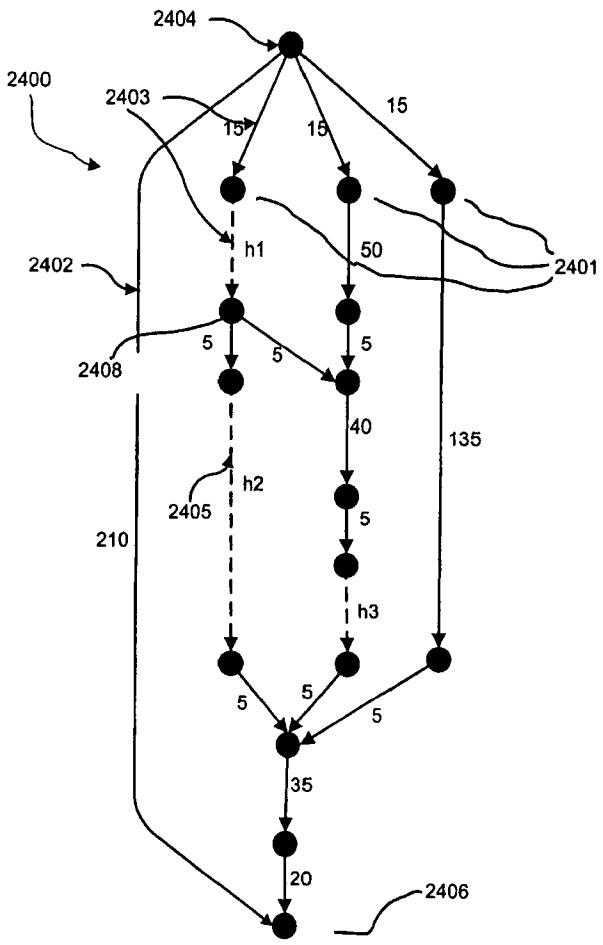
도면22


160 mm
2205
15 mm
15 mm
15 mm
2203
2217
55mm
5 mm
50 mm
5 mm
6 mm
2201
2209
5 mm
5 mm
5 mm
2221
135 mm
2219
40 mm
5 mm
5 mm
5 mm
2207
25 mm
5 mm
5 mm
2211
75 mm
5 mm
210 mm
5 mm
2223
5 mm
5 mm
5 mm
2213
35 mm
5 mm
5 mm
2216
2215
20 mm

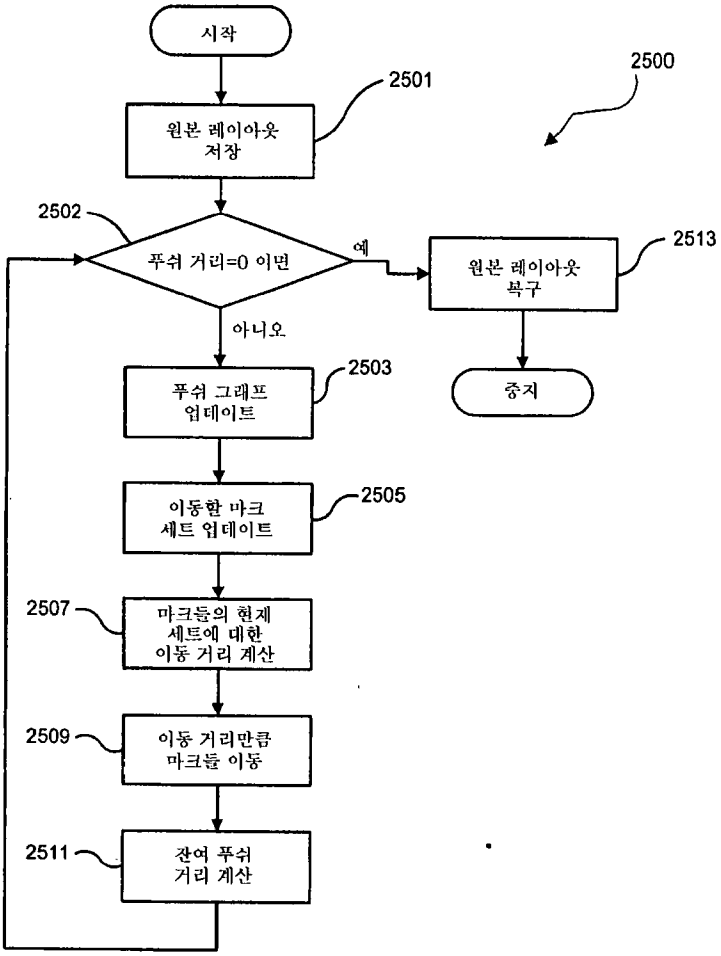
도면23



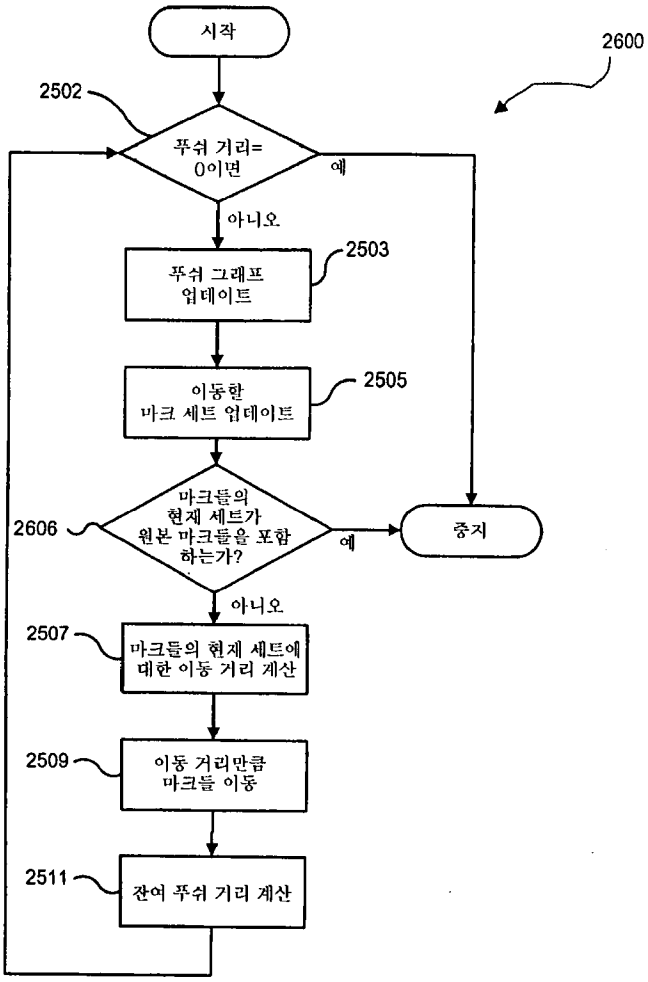
도면24


2404
2400
2403
15
15
15
h1
50
2401
2402
5
5
5
2408
40
135
h2
2405
5
210
h3
5
5
5
35
20
2406

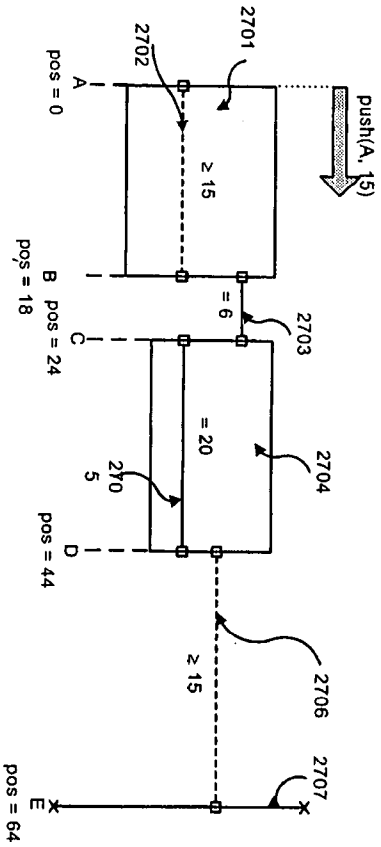
도면25


시작
2500
2501
원본 레이아웃 저장
2502
푸쉬 거리=0 이면
예
2513
원본 레이아웃 복구
아니요
중지
2503
푸쉬 그래프 업데이트
2505
이동할 마크 세트 업데이트
2507
마크들의 현재 세트에 대한 이동 거리 계산
2509
이동 거리만큼 마크들 이동
2511
잔여 푸쉬 거리 계산

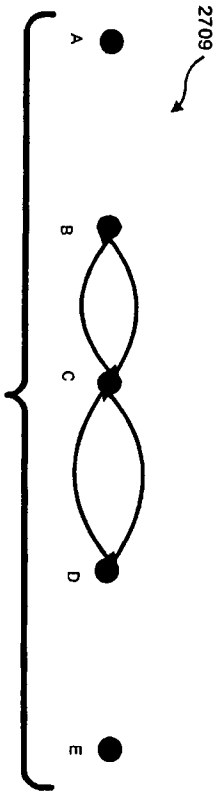
도면26


시작
2600
2502
푸쉬 거리=
0이면
예
아니오
2503
푸쉬 그래프
업데이트
2505
이동한
마크 세트 업데이트
마크들의
현재 세트가
원본 마크들을 포함
하는가?
2606
예
중지
아니오
2507
마크들의 현재 세트에
대한 이동 거리 계산
2509
이동 거리만큼
마크들 이동
2511
잔여 푸쉬 거리 계산

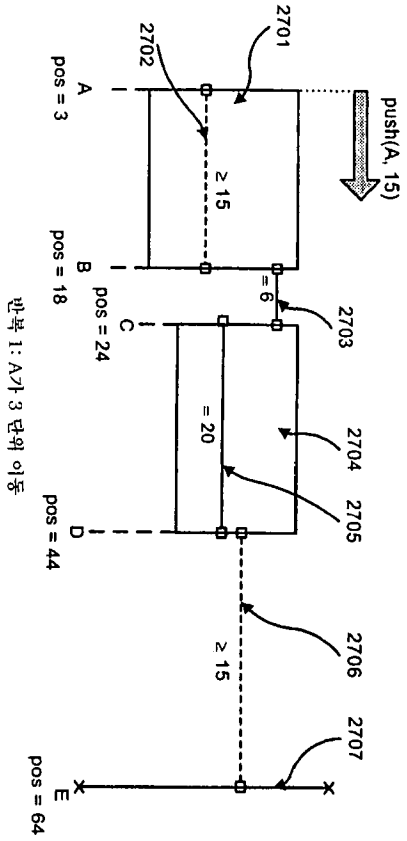
도면27a


2701
2702
A
pos = 0
push(A, 15)
≥ 15
B pos = 18
= 6
2703
C
pos = 24
= 20
2704
2705
D
pos = 44
≥ 15
2706
2707
E
pos = 64

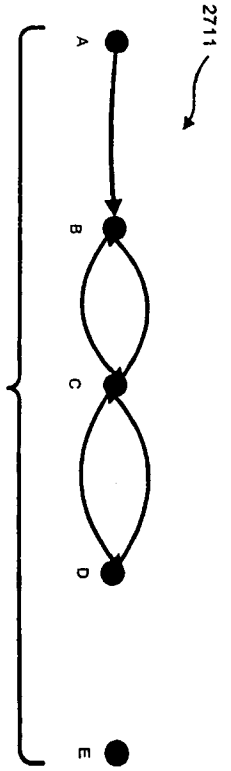
도면27b



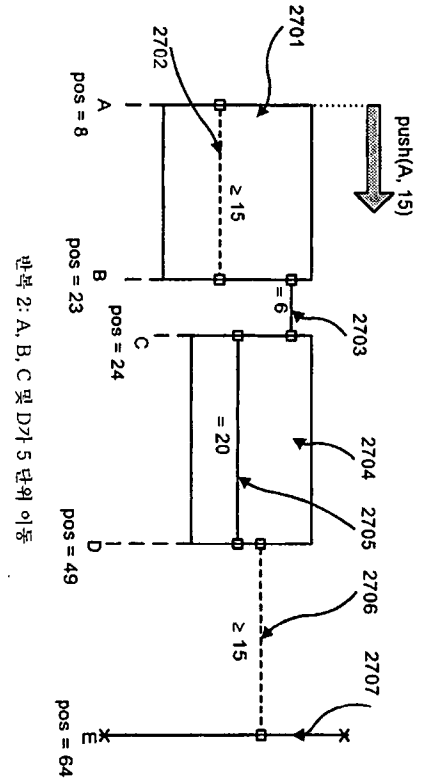
도면27c


2701
2702
push(A, 15)
A
pos = 3
≥ 15
B
pos = 18
= 6
2703
C
pos = 24
= 20
2704
2705
D
pos = 44
≥ 15
2706
2707
E
pos = 64
반복 1: A가 3 단위 이동

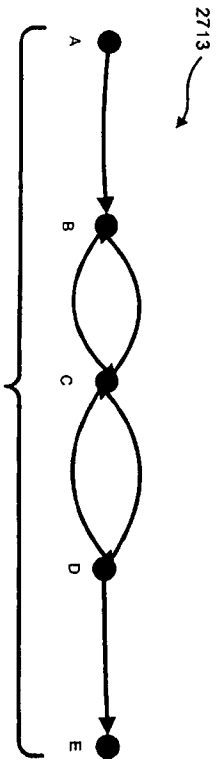
도면27d


2711
A
B
C
D
E

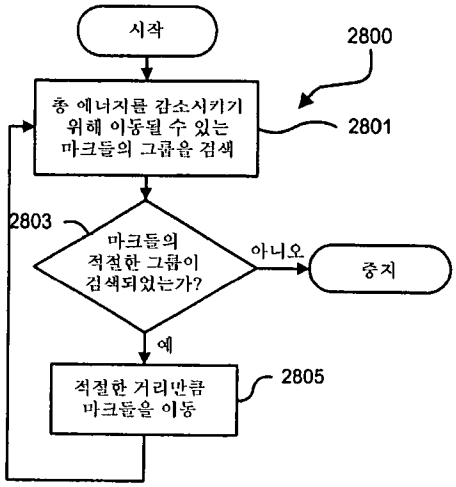
도면27e


2701
2702
A
pos = 8
≥ 15
push(A, 15)
B
pos = 23
= 6
2703
C
pos = 24
= 20
2704
2705
D
pos = 49
2706
≥ 15
2707
E
pos = 64
반복 2: A, B, C 및 D가 5 단위 이동

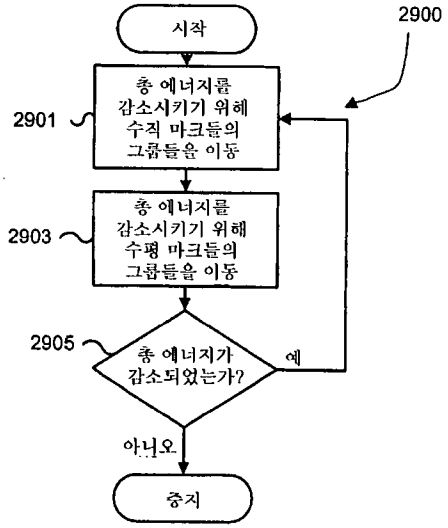
도면27f


2713
A
B
C
D
E

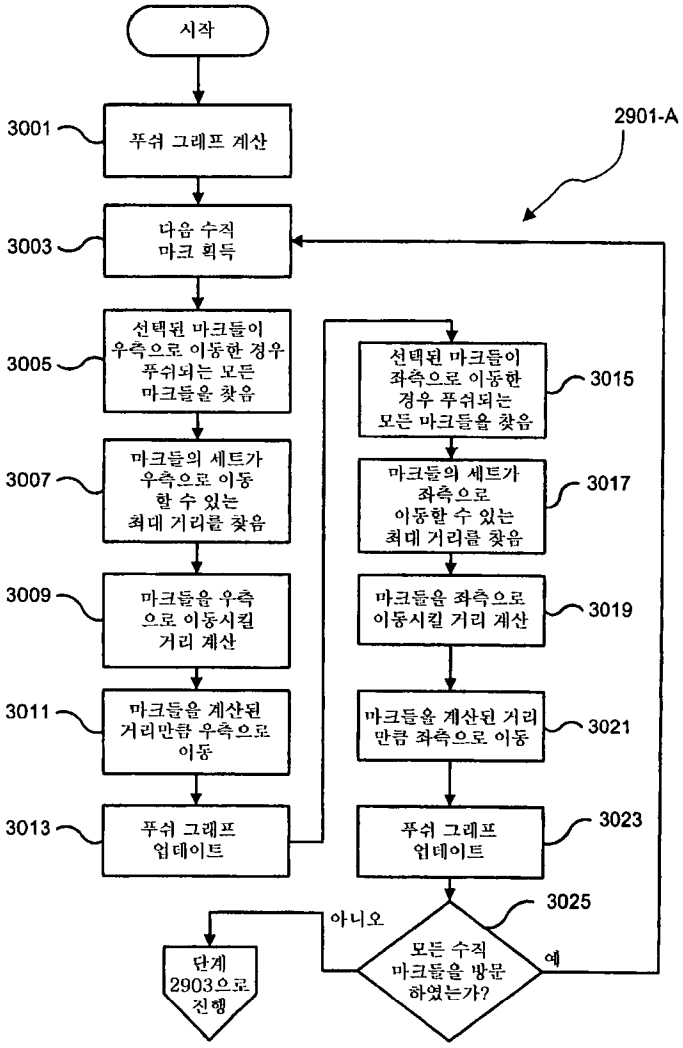
도면28


시작
2800
총 에너지를 감소시키기 위해 이동될 수 있는 마크들의 그룹을 검색
2801
2803
마크들의 적절한 그룹이 검색되었는가?
아니오
중지
예
적절한 거리만큼 마크들을 이동
2805

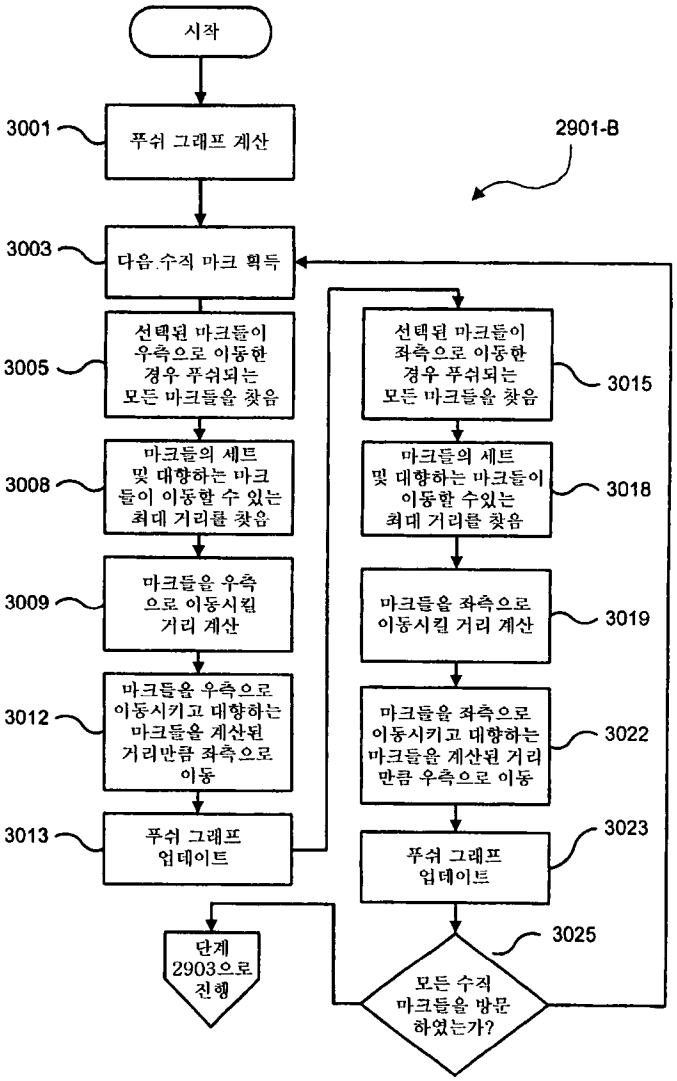
도면29


시작
2900
총 에너지를 감소시키기 위해 수직 마크들의 그룹들을 이동
2901
총 에너지를 감소시키기 위해 수평 마크들의 그룹들을 이동
2903
2905
총 에너지가 감소되었는가?
예
아니오
중지

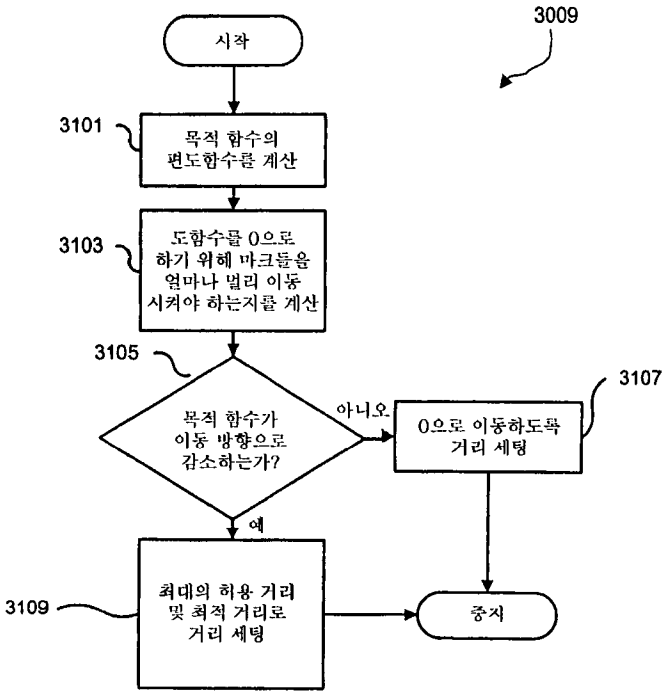
도면30a


시작
3001
푸쉬 그래프 계산
2901-A
3003
다음 수직 마크 획득
3005
선택된 마크들이 우측으로 이동한 경우 푸쉬되는 모든 마크들을 찾음
3007
마크들의 세트가 우측으로 이동 할 수 있는 최대 거리를 찾음
3009
마크들을 우측으로 이동시킬 거리 계산
3011
마크들을 계산된 거리만큼 우측으로 이동
3013
푸쉬 그래프 업데이트
3015
선택된 마크들이 좌측으로 이동한 경우 푸쉬되는 모든 마크들을 찾음
3017
마크들의 세트가 좌측으로 이동할 수 있는 최대 거리를 찾음
3019
마크들을 좌측으로 이동시킬 거리 계산
3021
마크들을 계산된 거리만큼 좌측으로 이동
3023
푸쉬 그래프 업데이트
3025
모든 수직 마크들을 방문 하였는가?
아니오
예
단계 2903으로 진행

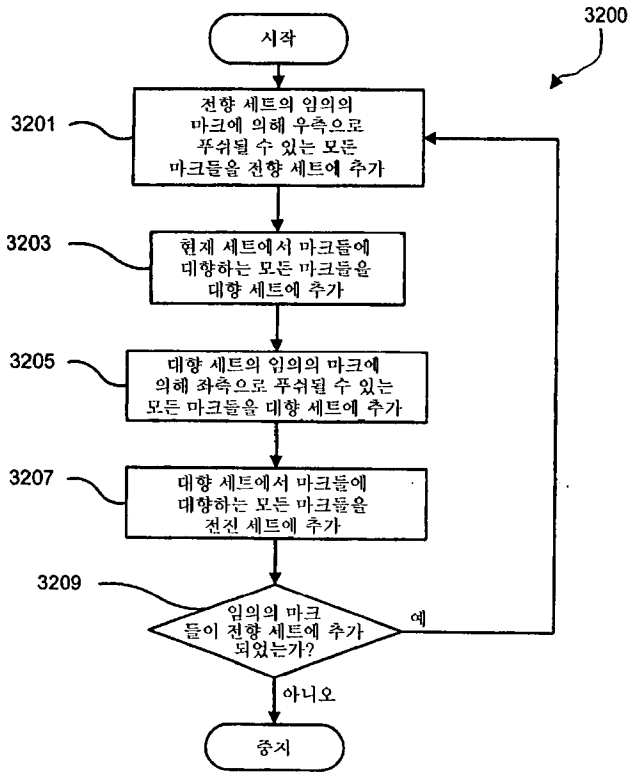
도면30b


시작
3001
푸쉬 그래프 계산
2901-B
3003
다음 수직 마크 획득
3005
선택된 마크들이 우측으로 이동한 경우 푸쉬되는 모든 마크들을 찾음
3008
마크들의 세트 및 대향하는 마크들이 이동할 수 있는 최대 거리를 찾음
3009
마크들을 우측으로 이동시킬 거리 계산
3012
마크들을 우측으로 이동시키고 대향하는 마크들을 계산된 거리만큼 좌측으로 이동
3013
푸쉬 그래프 업데이트
3015
선택된 마크들이 좌측으로 이동한 경우 푸쉬되는 모든 마크들을 찾음
3018
마크들의 세트 및 대향하는 마크들이 이동할 수있는 최대 거리를 찾음
3019
마크들을 좌측으로 이동시킬 거리 계산
3022
마크들을 좌측으로 이동시키고 대향하는 마크들을 계산된 거리만큼 우측으로 이동
3023
푸쉬 그래프 업데이트
3025
모든 수직 마크들을 방문 하였는가?
단계 2903으로 진행

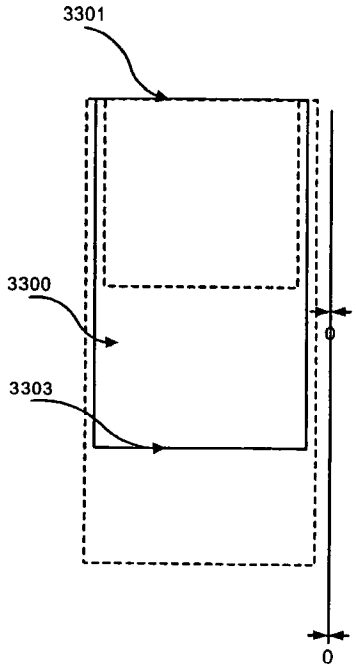
도면31


3009
시작
3101
목적 함수의 편도함수를 계산
3103
도함수를 0으로 하기 위해 마크들을 얼마나 멀리 이동시켜야 하는지를 계산
3105
목적 함수가 이동 방향으로 감소하는가?
아니오
3107
0으로 이동하도록 거리 세팅
예
3109
최대의 허용 거리 및 최적 거리로 거리 세팅
중지

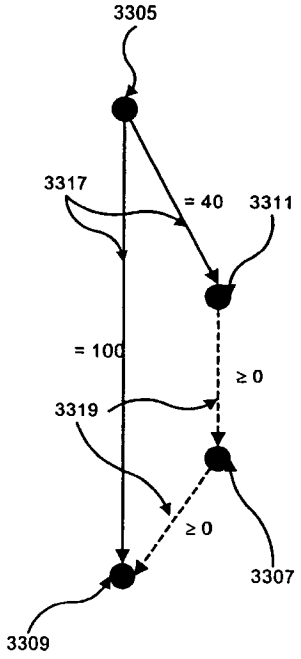
도면32


3200
시작
3201
전향 세트의 임의의 마크에 의해 우측으로 푸쉬될 수 있는 모든 마크들을 전향 세트에 추가
3203
현재 세트에서 마크들에 대향하는 모든 마크들을 대향 세트에 추가
3205
대향 세트의 임의의 마크에 의해 좌측으로 푸쉬될 수 있는 모든 마크들을 대향 세트에 추가
3207
대향 세트에서 마크들에 대향하는 모든 마크들을 전진 세트에 추가
3209
임의의 마크들이 전향 세트에 추가되었는가?
예
아니오
중지

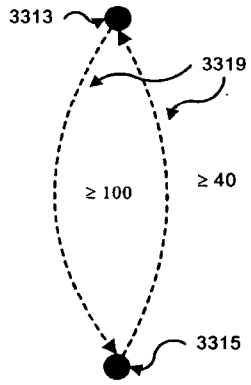
도면33a



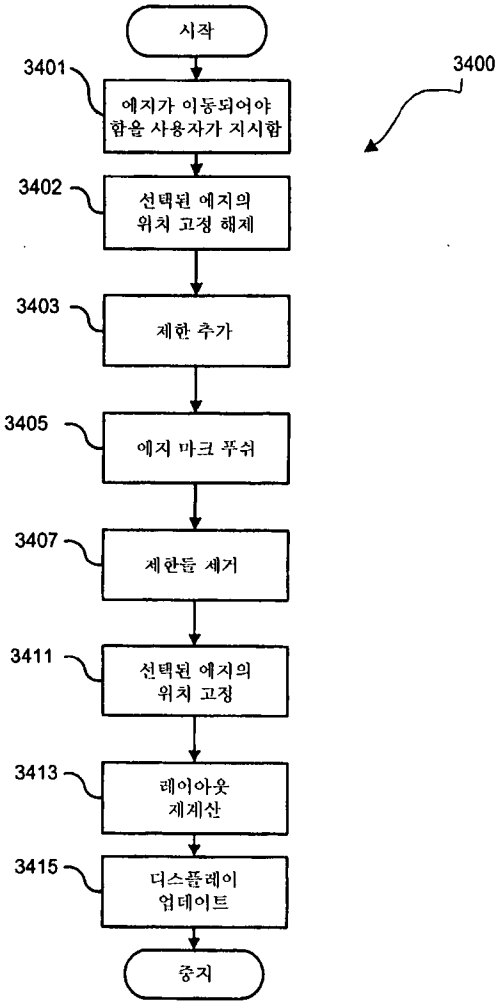
도면33b



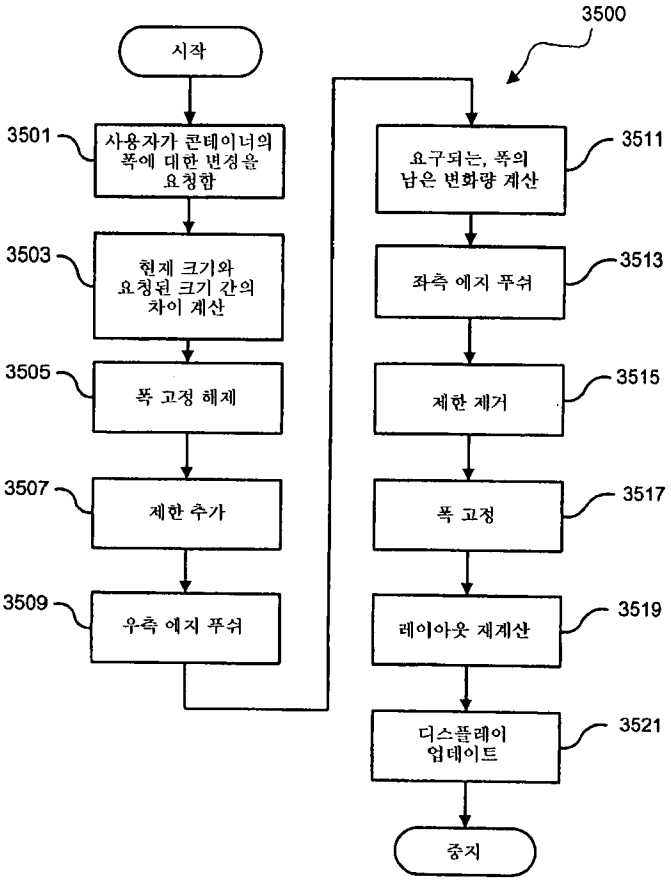
도면33c


3313
3319
≥ 100
≥ 40
3315

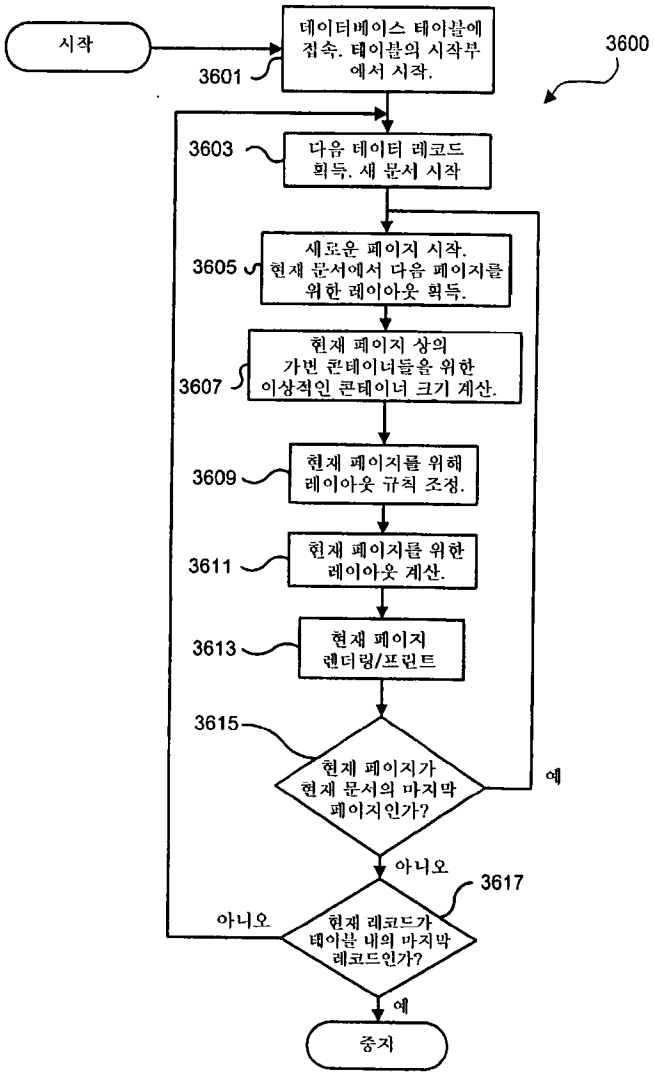
도면34


시작
3401
에지가 이동되어야 함을 사용자가 지시함
3400
3402
선택된 에지의 위치 고정 해제
3403
제한 추가
3405
에지 마크 푸쉬
3407
제한들 제거
3411
선택된 에지의 위치 고정
3413
레이아웃 재계산
3415
디스플레이 업데이트
중지

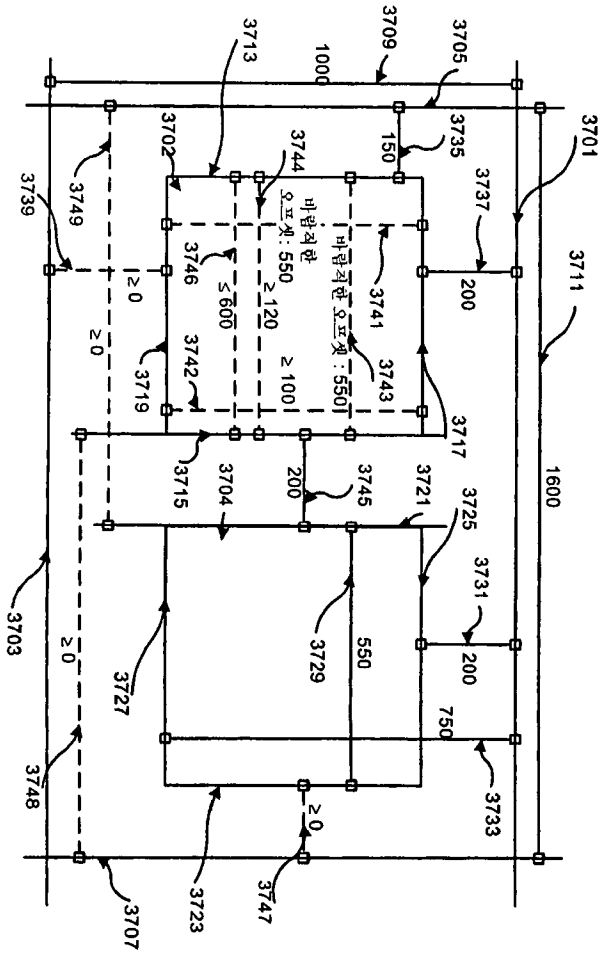
도면35


3500
시작
3501
사용자가 콘테이너의 폭에 대한 변경을 요청함
3503
현재 크기와 요청된 크기 간의 차이 계산
3505
폭 고정 해제
3507
제한 추가
3509
우측 에지 푸쉬
3511
요구되는, 폭의 남은 변화량 계산
3513
좌측 에지 푸쉬
3515
제한 제거
3517
폭 고정
3519
레이아웃 재계산
3521
디스플레이 업데이트
중지

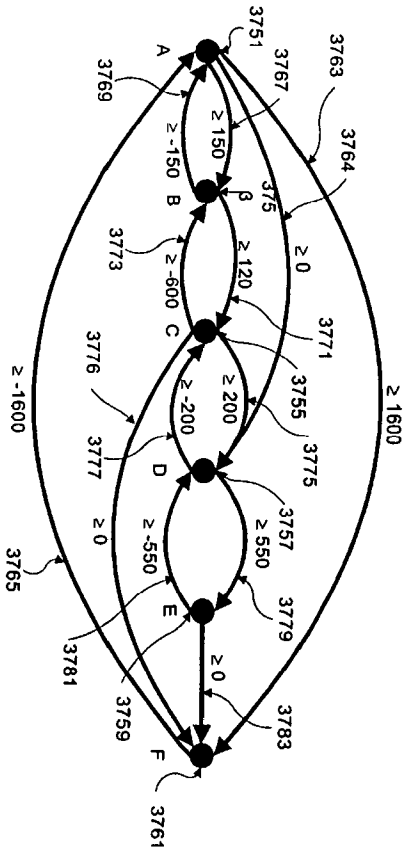
도면36


시작
데이터베이스 테이블에 접속. 테이블의 시작부에서 시작.
3601
3600
3603
다음 데이터 레코드 획득. 새 문서 시작
3605
새로운 페이지 시작. 현재 문서에서 다음 페이지를 위한 레이아웃 획득.
3607
현재 페이지 상의 가변 콘테이너들을 위한 이상적인 콘테이너 크기 계산.
3609
현재 페이지를 위해 레이아웃 규칙 조정.
3611
현재 페이지를 위한 레이아웃 계산.
3613
현재 페이지 랜더링/프린트
3615
현재 페이지가 현재 문서의 마지막 페이지인가?
예
아니오
3617
현재 레코드가 테이블 내의 마지막 레코드인가?
아니오
예
중지

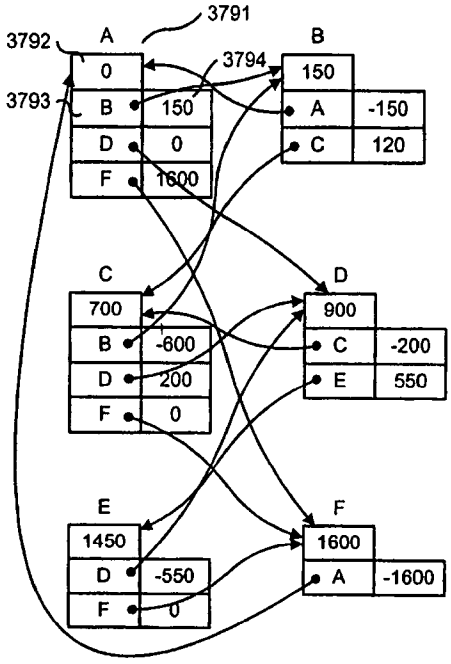
도면37a


3709
3705
3713
1000
3735
150
3701
3702
3744
3737
3739
3749
바람직한 오프셋: 550
바람직한 오프셋 : 550
3746
3741
200
3711
≥ 0
≤ 600
≥ 120
3742
3743
≥ 0
≥ 100
3719
3717
3715
3704
200
3745
3721
3725
1600
3731
3703
3729
550
3727
200
750
3748
3733
≥ 0
3723
3747
3707

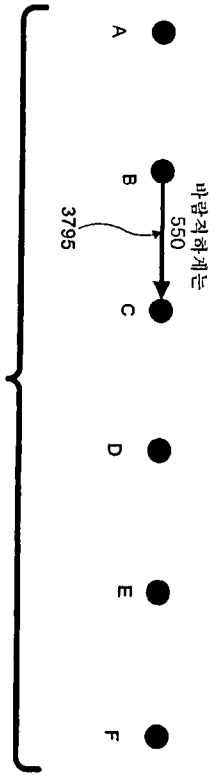
도면37b


A
B
C
D
E
F
3751
3763
3767
3769
3764
3773
3771
3776
3755
3775
3777
3757
3765
3779
3781
3759
3783
3761
≥ 150
≥ -150
≥ 120
≥ -600
≥ 200
≥ -200
≥ 550
≥ -550
≥ 0
≥ -1600
≥ 1600

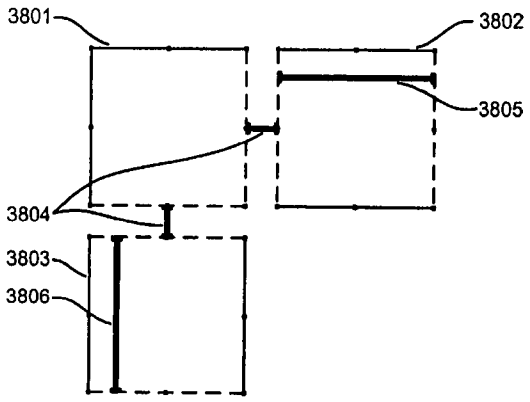
도면37c


3791
3792
3793
3794
A
0
B
150
D
0
F
1600
B
150
A
-150
C
120
C
700
B
-600
D
200
F
0
D
900
C
-200
E
550
E
1450
D
-550
F
0
F
1600
A
-1600

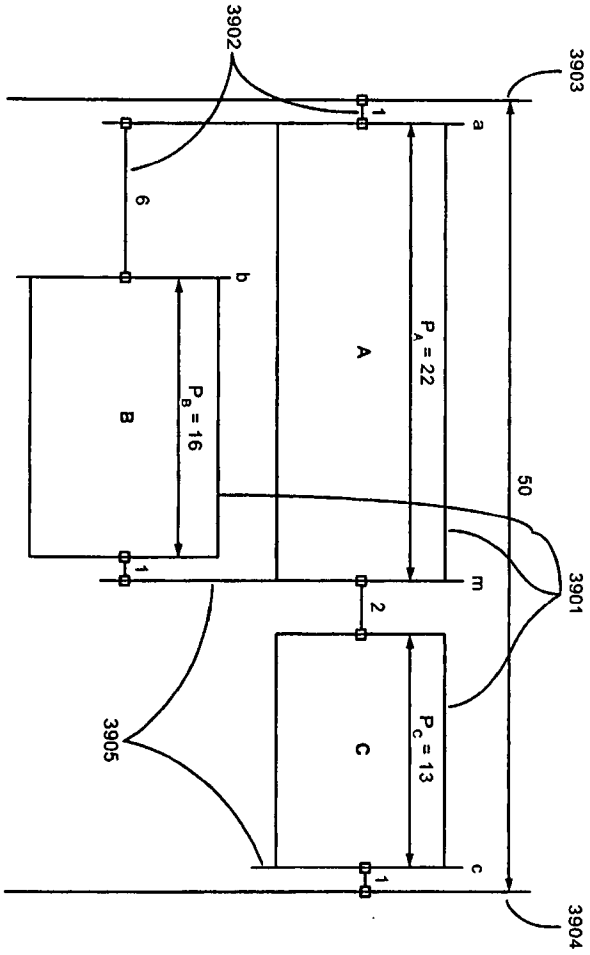
도면37d



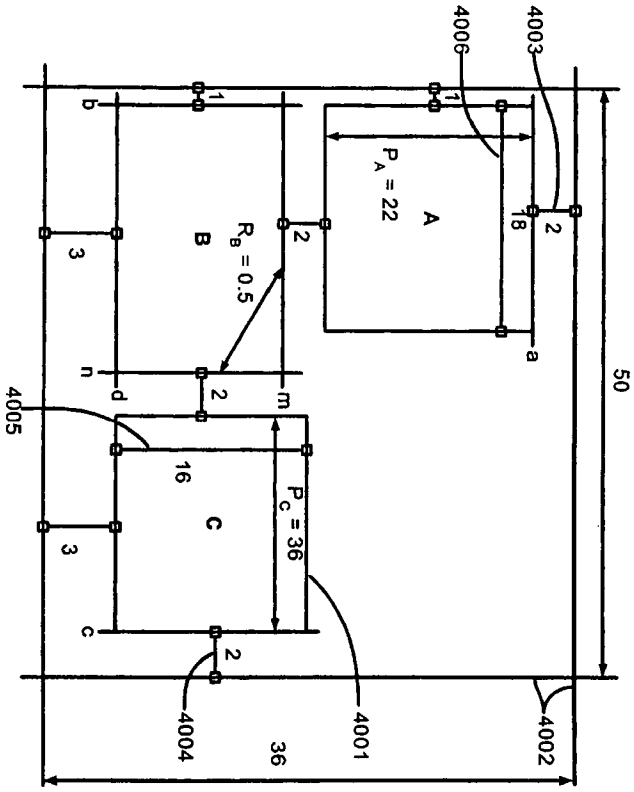
도면38



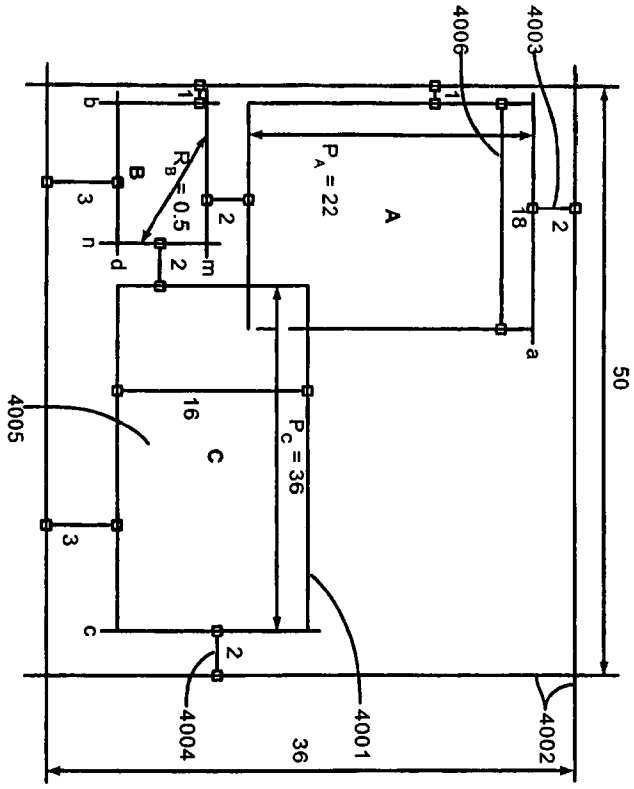
도면39



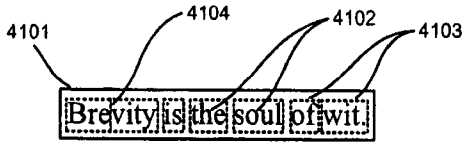
도면40a


4006
4003
1
1
b
A
P_A = 22
18
2
2
3
B
R_B = 0.5
n
d
2
m
a
50
4005
16
C
P_C = 36
3
c
2
4004
4001
4002
36

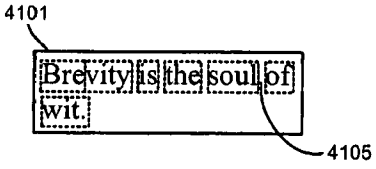
도면40b



도면41a

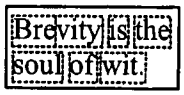


도면41b

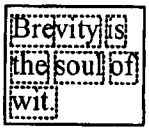


도면41c

도면41d



도면41e



도면41f

도면41g

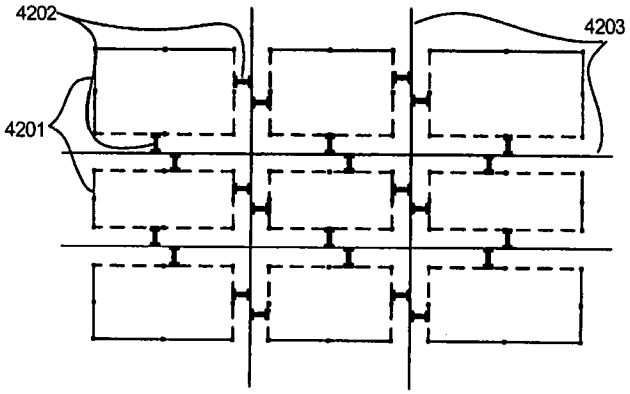
도면41h

도면41i

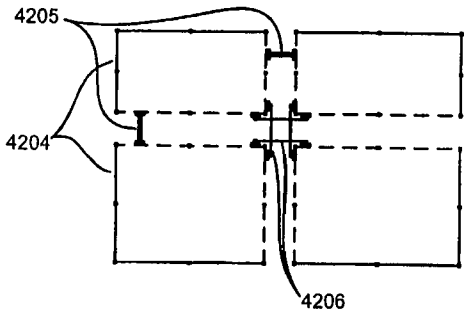
도면41j

도면41k

도면42a



도면42b



도면42c